HP PolyServe Software for Microsoft SQL Server: A highly available, flexible, simple approach to SQL Server consolidation white paper

How to achieve a 50% reduction in total cost of ownership with shared data clustering

The success of SQL Server in recent years has produced a phenomenon of SQL Server sprawl—the uncoordinated deployment of tens, hundreds or even thousands of database servers. The result has been low levels of utilization, poor availability and, most importantly, an enormous management burden inflicted on many IT departments. There is an acute need for a way to deploy SQL Server that allows capacity to be matched appropriately to need, that provides high availability without introducing extra complexity and extra cost, and that actually simplifies the life of administrators. This paper describes an approach that provides these benefits, which we call the HP PolyServe Software for Microsoft® SQL Server. The software supports both Microsoft SQL Server 2000 and SQL Server 2005. The software includes a component called Matrix Server, which is shared data clustering software. Shared data allows groups of servers and storage to work and be managed together flexibly to satisfy application requirements.

Introduction: The growth of SQL Server in the enterprise

Recent years have seen an explosion in the use of SQL Server. The proliferation of SQL Server and the cost-effective industry-standard servers on which it runs has brought tremendous benefits. Yet, this very success has also created ongoing problems for administrators responsible for its deployment and management. This white paper outlines an approach to SQL Server deployment and management that addresses these problems. We call this approach the HP PolyServe Software for Microsoft SQL Server.

The following sections cover:

- The typical patterns of SQL Server deployment in enterprises
- How these patterns have given rise to stereotypic problems that afflict administrators and raise costs
- How administrators have, to date, attempted to address these problems
- The ways in which these attempts have suffered limitations and difficulties
- How to overcome the problems of traditional SQL Server deployments, without the limitations of previous approaches, by building a flexible computing environment for SQL Server, based on HP PolyServe Software which leverages the shared data clustering capability
- How HP PolyServe Software provides the basis for managing a collection of servers and storage as a unit which can be applied to meet SQL Server capacity requirements, while also assuring high availability
- How HP PolyServe Software for Microsoft SQL Server resolves the problems of the traditional approaches
- Finally, how HP PolyServe Software for Microsoft SQL Server can
  - Drive up utilization, reducing server counts by 50%
  - Provide comprehensive, low-cost high availability
  - Simplify management, reducing administration time by up to 75%
  - Reduce the overall cost of implementing and maintaining SQL Server by as much as 70%.

Traditional SQL Server environments and SQL sprawl

SQL Server provides a high degree of functionality and performance at a low cost. It is widely supported by third-party packaged applications and by a broad range of development tools. It runs on industry-standard servers that have unmatched price performance. As a result, it is unsurprising that its use has grown as much as it has. However, the very attributes that have made it attractive have given rise to problematic patterns of growth.

- **One database, one server:** In many environments, administrators deploy a new server each time a database is required. When a packaged application is implemented that requires a database, a new machine is purchased and provisioned with SQL Server. When an in-house application is developed that requires a database, a new server is used. This one database/one server approach has the merit of simplicity, and the relatively low cost of SQL Server and the hardware on which it runs makes it feasible. However, as we discuss in the following section, the result is a great deal of complexity and ongoing operational expense. (In some cases, further servers are purchased to support development and test environments—one database, multiple servers!)

- **Proliferation at the departmental level:** Because SQL Server is relatively easy to buy and implement, it has often been purchased and deployed in an uncoordinated way at a departmental level, even in organizations that have well-organized central IT facilities. Over time, the number of servers dedicated to running SQL Server can grow to an astonishing degree, with correspondingly high ongoing costs.

- **Inadequate provision of high availability:** There are well-established ways to implement high availability for SQL Server databases, typically using failover clustering in active-passive server pairs. However, as we discuss in the following section, these approaches introduce significant extra cost and much complexity compared to non-clustered SQL Server. The result, in many cases, is that the threshold required to justify a highly available implementation has been quite high. Consequently, it is commonplace to find databases that in fact are important for continuing business operations but are unprotected by high availability. This is a further consequence of the relatively uncoordinated deployment of SQL Server.

Together, these creeping patterns have created what many IT managers are calling SQL sprawl: widespread, relatively uncoordinated and/or unplanned database deployments, typically in the form of many individual servers and a smaller number of active-passive server pairs clustered for high availability. Depending on the size of the organization, **SQL sprawl** may affect tens, or hundreds, or even thousands of servers. Regardless of the absolute numbers involved, SQL sprawl represents a waste of resources and an ongoing burden on the IT infrastructure. The next section teases out some of the costs imposed by SQL sprawl.

Figure 1. Microsoft SQL Server sprawl
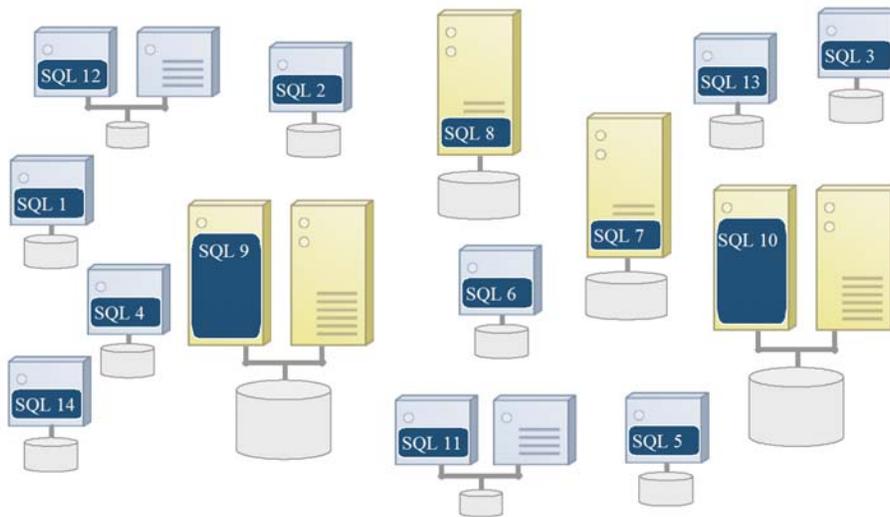
The costs of SQL sprawl

Low levels of utilization

The one database/one server approach to database deployment has, inevitably, produced appallingly low levels of utilization. In our experience working with customers, it is not uncommon to find most database servers running at **5–15% CPU utilization**—particularly with modern server hardware. Of course, low levels of utilization imply higher-than-necessary capital costs, or, looking at it another way, low levels of utilization mean there is a large amount of stranded capacity in the environment that could be put to better use.

While low utilization levels arise in some cases because of the simplicity of deploying a new server for each database, other factors can be at work. In a typical environment, **the penalty for under-provisioning a machine for SQL Server is high**. This often causes a last-minute procurement of a larger and faster server as a replacement and a weekend or evening spent configuring the new machine, copying the database to it, and testing that it works properly. Figure 2 illustrates just how painful this can be. To avoid this, it is simpler to buy a large server up front, even though that inevitably produces large amounts of stranded capacity.

Figure 2. The traditional process of moving a SQL Server instance from one server to another

Traditional Database Migration Starting Point: New server loaded with Windows®	
21:00	**Document SQL parameters and directories on Old Server**
21:00	Make current backup of all databases
21:15	Shut down database on Old Server
21:20	Copy the SQL Server folder to New Server
21:22	Export SQL Server registry keys
21:25	For each user database: • Detach database from source instance • Copy database to destination instance disk • Attach database to destination instance disk
22:25	Move msdb database to New Server
22:45	Move master database to New Server
23:05	Move the model database to New Server
23:25	Copy the registry keys to New Server
23:30	Backup
23:45	Install SQL Server on New Server (parameters and directory structures must be the same)
00:45	Install service packs and hotfixes to New Server
01:15	Stop SQL Server on New Server
01:20	Export SQL Server registry keys (backup)
01:25	Copy the SQL Server folder on New Server (backup)
01:35	Copy SQL Data from Old Server to New Server over network (including the entire instance folder and all databases)
02:15	Shut down and power off Old Server
02:20	Import the SQL Server registry keys on New Server
02:25	Change New Server IP address to Old Server IP address
02:30	Change New Server name to Old Server
02:35	Reboot New Server
02:45	Start SQL Server on New Server
03:00	Test SQL Server and connectivity to ensure successful move
03:10	Complete

Storage utilization suffers for a similar reason. Running out of storage on a database server may require a laborious and expensive process of procuring and then installing new disks or RAID arrays. Because installing more storage can require downtime, this again may need to be done in an off-hours session. It is simpler just to purchase more storage at the start of the project. However, the result across many servers can be a huge amount of **stranded, unused storage capacity**.

In some cases administrators have chosen to deploy databases on dedicated servers to provide strong **performance isolation** from each other and from other workloads. In a traditional environment, moving a database from one server to another may require hours of evening or weekend work; as a result, the penalty for allowing a server to become overloaded is high. Giving each database its own server is the safest option, but one that is ultimately expensive.

Finally, sometimes individual servers are chosen to minimize the potential **exposure from server failure**. This rationale reflects the perception that high availability is expensive and complicated. In fact, that belief itself merits further investigation as a major source of problems in traditional SQL deployments.

Inadequate availability

In many traditional environments, **few SQL Server databases are protected by high availability**. Typically, this springs from a belief that high availability must impose significant extra expense and complexity.

Although there are multiple ways to provide high availability for SQL Server, the most common approach is to use a pair of servers configured as an active/passive failover cluster. Both servers in the pair are connected to storage that contains the database. In normal operation, only the primary server has access to this storage, and it hosts the database. If it fails, the secondary takes over control of the storage, mounts the appropriate drive letters, and starts up the database.

Obviously, in this approach, **failover clusters require twice as many servers** as unclustered deployments, and therefore inflate hardware costs. Some clustering approaches require that servers to be clustered use identical hardware; as a result, moving an existing database to a cluster may require purchasing not just one extra server, but a fresh pair. Although it is possible to set up a failover cluster so that each server runs a database during normal operation and provides backup for the other server in the event of failure, this is not always done. The net effect is that traditional fail-over clustering very often not only raises costs but also depresses the overall level of server utilization.

Furthermore, **clustering traditionally introduces administrative complexity**. At a mundane level, setting up a failover cluster requires the administrator to ensure that the drive letters used by a database on one server are not in use on the other. It also may be required to ensure that the two servers are at the same OS patch level.

A more subtle issue—and therefore one potentially more fraught—is the administrator must ensure storage is configured to allow the hand-off between servers involved in a failover. Disk transition during failover may involve storage features like SCSI RESERVE and RELEASE commands that are not commonly used in the same way during normal operation. As a result, a cluster may work well in normal operation but not, in fact, have been configured correctly for a failover event. The result can be so-called failed failovers.

The net effect of these issues is that many administrators perceive high availability for SQL Server to entail a level of expense and complexity significantly in the preceding section non-clustered deployments. In turn, this leads to relatively few SQL Server databases being protected, and to consequent ongoing downtime costs.

Management overhead

More servers mean more work. Studies have shown that initial hardware and software acquisition costs represent a small part of lifetime server total cost of ownership (TCO). A large part of the balance consists of the cost of managing and maintaining that server after it is deployed.

Consider **operating system patches and upgrades**. Even with automated processes, each incremental server in an environment represents an upgrade/patch point that needs to be validated and, if the patch fails, investigated. Of course, upgrades or patches may entail downtime (and ironically, in some cases, patching a clustered server pair can be more complicated than patching an unclustered server).

Storage management and related issues represent a significant ongoing burden for each deployed server. Each server has its own backup job; someone needs to validate that each of these jobs is succeeding and investigate if not. Each server has a free space pool that must be monitored. As discussed in the preceding section, clustered servers have particular storage management needs of their own; to ensure failovers will succeed when required, it may be necessary to verify the storage configuration has not been altered inadvertently subsequent to initial set-up.

As the number of servers dedicated to SQL Server grows, these ongoing management responsibilities can consume a significant amount of administrator time and, therefore, cost. Over the years, computing power has declined in cost, but the time of talented and well-trained administrators has become more valuable. Thus, in many organizations, **the greatest cost of SQL Server sprawl is the burden it places on administrators to perform ongoing maintenance chores, fight fires and focus on reactive tasks**.

Initial responses to SQL Server sprawl

Unsurprisingly, given the costs and problems associated with SQLServer sprawl, many organizations have launched efforts to address the problem. Of course, simply adopting an organized, systematic approach and implementing best practices as SQL Server moves into the data center can achieve some benefits.

On the other hand, the very fact of bringing SQL Server into a more organized environment can throw into sharp relief the waste of resources caused by low utilization levels, the failure to provide adequate high availability across a broad spectrum of databases, and the ongoing burden of administering many servers. As a result, people have sought out alternative approaches for deploying SQL Server in an attempt to address these concerns. Unfortunately, many of these suffer limitations and drawbacks of their own.

Storage area networks

As SQL Server moves into the data center, implementers are shifting away from traditional direct-attached storage to storage area networks (SAN). Moving database storage to a SAN can help address some, but not all, of the storage management issues related to databases. On a SAN, more storage can easily be allocated to a server running out of space, making provisioning dramatically easier than in a traditional environment with direct-attached storage. A SAN also provides a base for implementing high availability, since servers are now physically connected to the same storage.

Unfortunately, moving to a SAN on its own does nothing to solve the other problems described in the preceding section. The fundamental limitation of a typical SAN deployment is that while the storage is physically consolidated, there is no logical consolidation of server data. In particular, each server has its own partition of the storage on the SAN that it alone can access, as shown in Figure 3. The result is that servers and their associated data still exist as independent silos and must be managed separately. Server utilization remains low, availability remains challenging and complex to implement and management of many servers continues to be taxing and laborious.

SANs help by making storage provisioning easier, but are not a complete solution.

Figure 3. In a traditional SAN deployment, each server has its own area of storage
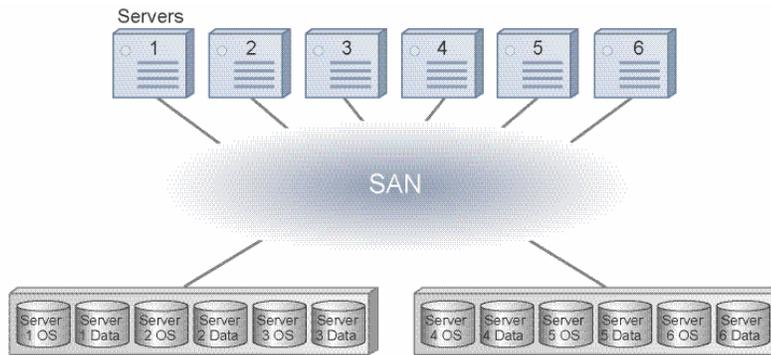
Consolidation on large servers

One obvious approach to taming SQL sprawl, perhaps in conjunction with SAN adoption, is to move databases from existing machines onto a smaller number of much larger servers. Each larger server then handles multiple databases simultaneously.

On the positive side, this can reduce the number of servers that need to be administered, with a corresponding reduction in power and cooling requirements, the number of operating system images that need to be patched and updated, the number of backup jobs in the environment, and the amount of attention paid to monitoring free space. However, it comes at a cost:

- **Large servers are expensive.** Disproportionately expensive, in fact: the price/performance of 2- and 4-way servers is far better than that of 8-way servers and in the preceding section. Thus, it is significantly more expensive to buy a small number of very large servers to handle a given set of databases than to buy a larger number of standard 2-way or 4-way servers for the same workload.

- **Existing hardware is not reused.** Moving to new hardware means abandoning existing servers. Ironically, if other uses for these servers cannot be found, the overall utilization level of all purchased hardware can go down.

- **High availability becomes more vital, but is still complicated and expensive.** With more databases on each server, it becomes all the more pressing to provide high availability to protect against hardware failures. However, if a second server is required, you have now doubled the cost of an already expensive server and cut utilization in half. High availability also introduces higher levels of complexity for server and storage administration.

- **Over-consolidation is painful to fix.** For this approach to be successful, it makes sense to put as many databases as possible on each large server. However, if a server then becomes overloaded and performance suffers, it can be painful and laborious to fix the problem: typically it requires an off-hours session to copy one or more databases off the overloaded server onto a server with spare capacity and verify that it works in that new configuration. To avoid this, it may be tempting to ensure each large server has ample spare capacity—but doing that, of course, means stranding capacity, and stranding capacity that in this approach is purchased at a high cost.

Straightforward consolidation on large servers has one very appealing benefit: it can reduce the number of moving pieces in the data center devoted to running SQL Server and thus, potentially, diminish overall management effort. However, it does so at the cost of shifting from more cost-effective hardware to less, and it may actually increase the complexity of each server installation, due to the need to provide high availability and to ensure workloads are matched correctly to server size.

Server virtualization

Another approach uses server virtualization technology to run multiple virtual machines within a single physical server. Server virtualization has been effective in recent years in consolidating servers that run miscellaneous small, lightly loaded applications. These dedicated servers can be turned into virtual machines on many fewer physical servers. This works even if the applications require different operating system patch levels, since each virtual machine has its own copy of the operating system. This kind of consolidation can save data center space and reduce power and cooling requirements without changing in any way the management of the applications themselves.

Virtual machines are also widely used in testing and development, where it is helpful to be able to maintain multiple complete images of target environments—including, for example, specific operating system patch levels—without having to have physical hardware for each one. In general, server virtualization is able to reduce the number of physical servers required to host a given number of virtual OS environments and can actually save hardware costs if it removes the need to purchase extra servers.

Finally, some server virtualization products allow virtual machines to be moved easily from one physical server to another. This capability would appear to be an appealing basis for flexibly matching SQL workloads to the servers—though, as we discuss in the following section, this is often not the case.

Despite these successes, however, server virtualization has severe limitations as a basis for hosting production SQL Server databases:

- **No simplification of software administration tasks.** Unlike straightforward consolidation onto large servers as described in the previous section, server virtualization does not simplify software administration and maintenance. Consider consolidating 50 physical servers running SQL Server onto five to 10 physical servers with 50 virtual machines. When a security patch or SQL hot fix comes out, there are still 50 updates required. There are still 50 backup jobs that need to be configured and monitored, and 50 places to monitor for free space exhaustion. All the software-related administration chores that existed before remain after the consolidation is complete. While having individual OS images in each virtual machine is a benefit when you are consolidating miscellaneous applications that require their own particular OS environments, it is a tremendous missed opportunity with a more homogenous workload like SQL Server.
- **Large servers may be required.** On the other hand, like straightforward consolidation, server virtualization may require large servers to have enough performance to handle SQL workloads. This requirement derives partly from the next problem:
- **Virtualization imposes a performance overhead.** In a virtual machine environment, the operating system does not interact directly with native hardware. Instead, all interactions are intercepted and mediated through a software virtualization layer. This imposes a performance penalty, especially on I/O intensive applications like SQL Server.
- **Virtualization limits the resources that can be directed to a given virtual machine.** While it is possible to run server virtualization products on servers with many CPUs and many gigabytes of RAM, today there are severe limitations on how many CPUs and how much memory any given virtual machine can use. For example, one popular product limits virtual machines to just two CPUs and less than 4 GB of RAM. Thus, any database that might grow to need more power than this cannot be put into a virtualized environment.

- **Virtualization elevates the importance of high availability without making it easier.** In a virtual machine environment, it becomes even more important to have high availability protection, since now a single physical server failure could take down many databases. In this respect, virtualization resembles straightforward consolidation as discussed in the preceding section, but it is actually makes high availability more complicated to achieve. Instead of having to install clustering protection once for each large server, in the virtual machine approach, clustering has to be installed into each individual virtual machine. In fact, it may be necessary to create new virtual machines on other physical servers to provide fail-over targets, meaning the number of software environments to be managed may actually double by moving to a virtualized environment.

- **Some virtualization features may be incompatible with high availability.** For example, consider the VMotion capability of the popular VMware virtualization products. VMotion allows a virtual machine to be moved from one server to another. VMotion itself is not a high availability feature: both the origin server and the destination must be up and healthy throughout the migration process; if a server fails, it is too late to use VMotion to move its virtual machines. However, beyond this, VMotion is actually incompatible with high-availability failover for a virtual machine. That is, an administrator must decide, ahead of time, whether a given virtual machine will be used with VMotion, in which case clustering software cannot be installed within the virtual machine, or whether the virtual machine will include clustering software, in which case VMotion cannot be used on that machine. As a result, VMotion is far less useful in a production SQL Server environment than it would otherwise be.

- **Virtualization may require administering non-Windows operating systems.** For example, VMware's high-end virtualization product, called ESX, is based on a Linux administration console that runs as a virtual machine on each server. So, even in an environment that is otherwise all Windows, this form of virtualization requires administrators to accept responsibility for Linux operating system instances.

- **Supportability.** In some cases, running mission-critical applications like SQL Server within a virtualized environment may raise obstacles to support. Microsoft's support policy states "Microsoft does not test or support Microsoft software running in conjunction with non-Microsoft hardware virtualization software." Microsoft Knowledge Base article #897615, October 27th 2005. It further states that for any Microsoft product running in a non-Microsoft hardware virtualization product, it may be required to reproduce any issues in a non-virtualized environment to obtain support.

For all these reasons, server virtualization represents a poor solution to the problems of SQL sprawl. The performance limitations alone rule it out for many databases. However, even if the performance problem could be overcome, server virtualization fails to provide a solution for the high-availability needs of a SQL environment and, most importantly, offers no relief from the ongoing administrative burden that is SQL sprawl's worst affliction.

A better solution: HP PolyServe Software for Microsoft SQL Server

What is desired is a way to host SQL Server that offers the benefits of the preceding approaches, without their limitations, and comprehensively addresses the problems of SQL sprawl: low levels of utilization, poor availability and ongoing management burden.

The ideal solution would include:

Table 1. HP PolyServe Software for Microsoft SQL Server ideal solution

Like Storage Area Networks alone	But…
• Simplified Provisioning of storage to servers and higher storage utilization	• Also a reduced number of backup jobs and a reduced number of free space pools to monitor
• A basis for high availability	• An easier way to deploy high availability, without complex and brittle storage configurations
Like consolidation on large servers	**But…**
• A reduction in the number of servers and the number of OS and software environments to be managed	• Reusing existing hardware, without requiring new, large, and expensive servers
• High levels of server utilization	• Without imposing a steep penalty if consolidation leave a server overloaded
Like server virtualization	**But…**
• A simple way to shift workload flexibly from one server to another	• Complete compatibility with high availability, no performance compromises and no support headaches

Unlike any of these approaches

A simple way to provide high availability universally to all databases, without doubling hardware

A way to perform hardware and software maintenance without incurring downtime

Shared data makes it possible

The fundamental idea of the HP PolyServe Software for Microsoft SQL Server is to treat the hardware resources—servers and storage—that support a collection of SQL Server databases as a single pool that can be directed, as required, toward any database. Figure 4 illustrates the basic design.
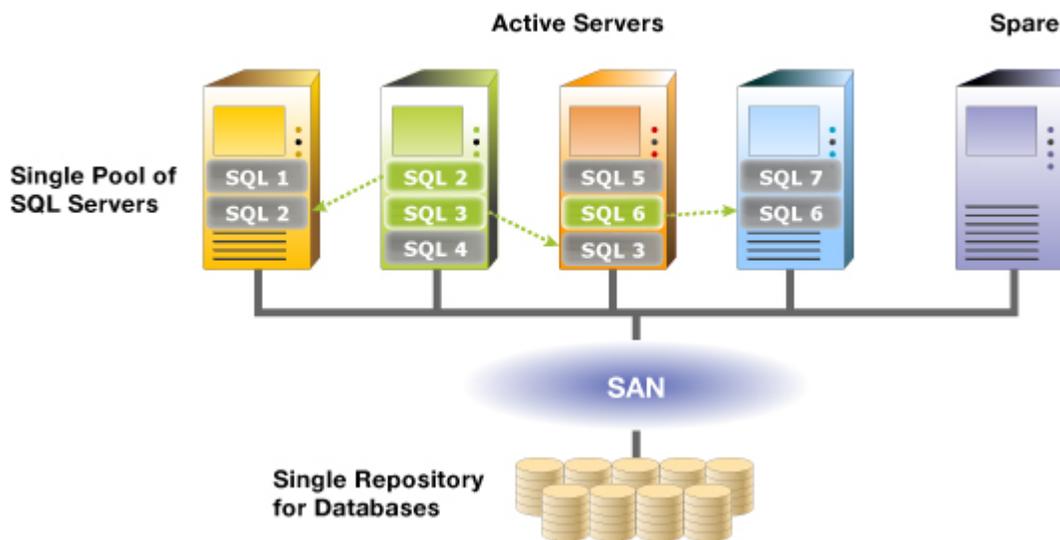
The key point of Figure 4 is that the files containing the data for all the databases are stored in a single place that can be accessed by all of the servers simultaneously. This capability is called shared data clustering. It brings the following fundamental benefits:

- **A single pool of servers:** In this approach, you no longer think of installing a database on any particular server. Instead, you install into the cluster, and the database can then run on any server in the cluster. This allows an administrator to move a database from one server to another to rebalance load and maintain an appropriate level of utilization—without any need to copy or migrate data. It also means if any server fails, the databases it had been running can immediately and automatically be restarted on other machines, ensuring high availability.

- **A single pool of storage:** Similarly, there is no need to manage storage on a server-by-server basis and no requirement to have a backup job per server or separate monitoring for each server's free space pool. In addition, because the servers are connected to storage over a SAN, it is easy to provision more storage when required—but that is only done when the environment as a whole needs more space.

- **Flexibility:** A shared data cluster can be formed from servers you already own. There is no need to buy matched servers; you can mix servers from different vendors, with different processor types and speeds, different numbers of processors and different memory configurations.

- **Easy scalability:** It is easy to add servers to a cluster if overall demand grows, and databases can be shifted in a matter of seconds onto newly added servers with no need for data copying. (Conversely, if workloads drop, you can shift databases off some of the servers and remove them for repurposing.) Databases receive full native performance with no virtualization overhead or virtualization limits, so if a database requires the full speed and capacity of a large server, it can have it.

- **Easy high availability:** Because all servers have access to all databases, high availability becomes easy to implement with no requirement for doubling up hardware. Simply by specifying where a database should be restarted in the event of failure—from among any of the servers in the cluster— you can ensure the database will remain available. The shared storage pool requires none of the complicated and brittle configuration on a server-pair-by-server-pair basis that traditional failover clustering can entail.
- **Easy migration to SQL Server 2005:** HP Technical Services offers an easy, five-step migration to SQL Server 2000 or SQL Server 2005. HP PolyServe focused consultants will work closely with your technical team to ensure a successful migration to SQL Server 2000 or 2005 running in HP PolyServe Software for Microsoft SQL Server.

The following sections describe in more detail the components of the HP PolyServe Software for Microsoft SQL Server and how it offers the benefits listed in the preceding section.

HP PolyServe Software for Microsoft SQL Server

HP PolyServe Software for Microsoft SQL Server is an integrated product that allows a collection of servers and SAN storage to be managed as a single entity for hosting many SQL Server databases. It is made up of several components that work together:

- Matrix Server, which provides shared data clustering and allows a set of servers to be managed as a unit.
- Volume Manager, which allows storage from multiple arrays to be used and managed as a unit.
- HP PolyServe Software for Microsoft SQL Server, which adapts the core shared data clustering capabilities of the previous two components for use with SQL Server.
- Support for SQL Server 2005. HP PolyServe Software supports both SQL Server 2000 and SQL Server 2005, running in the same cluster.

The following sections describe each of these in turn.

Figure 5. A typical Matrix Server deployment

Storage Arrays with Single Pool of Data
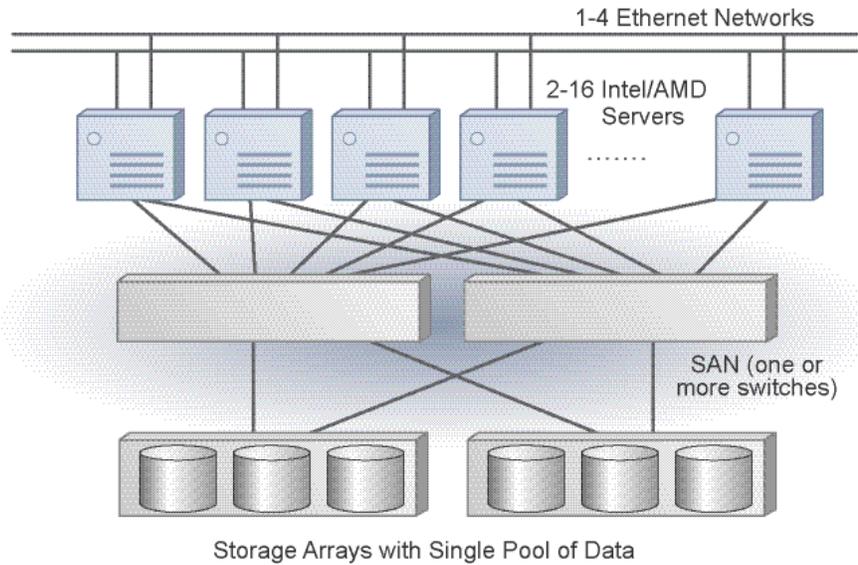
Matrix Server

Matrix Server runs on Windows Server 2003 and Windows Server 2003 R2 and allows up to 16 servers to form a shared data cluster, with a single storage pool and a single management point for applications. A basic block diagram of a typical deployment is shown in Figure 5. A Matrix Server environment consists of Intel®- or AMD-based industry-standard servers or blades, a storage area network (connected by way of Fibre Channel or iSCSI), one or more storage arrays attached to the SAN, and one or more standard Ethernet networks linking the servers. A Matrix Server cluster can include any mix of servers and storage from different vendors with different specifications—all that is required is that they all share standard SAN and Ethernet connections and run standard Windows operating system environments. After the server and storage hardware is configured and the operating system is installed, Matrix Server installation is as easy as installing an .MSI file and answering a few questions.

Matrix Server itself includes these capabilities:

- A cluster file system that allows multiple servers to access the same storage simultaneously
- A high-availability and application control engine that monitors hardware and software health and can restart and move applications within the cluster
- A single management console called Matrix Manager that allows the status of all hardware and software resources in the cluster to be viewed at a glance and controlled from one location.
- Support for industry-standard servers (or blades) and storage. Use 32- or 64-bit Intel Opteron or Intel E64T servers, of any configuration. Utilize industry-standard storage connected by way of Fibre Channel (FC) or iSCSI.

Cluster file system

The core of the Matrix Server approach is a symmetric, high-performance, robust, native, NTFS-compatible cluster file system. This file system allows multiple servers to access files located on storage connected to a common SAN simultaneously with full integrity. Typical SAN deployments in the past—without shared data clustering software like Matrix Server—allowed multiple servers to use a single common storage array, but only if that array was partitioned into separate pieces and each piece was used by only one server at a time, as shown in Figure 3. With Matrix Server, the files in an array can be accessed by all the servers with no need for partitioning.

Matrix Server's cluster file system is, as described in the preceding section:

- **Symmetric:** Each server in the cluster accesses disk storage directly, with no single file server or master server mediating access. If the fourteenth server in a cluster needs to create a file in the shared space, it can create that file itself, without waiting on any particular other server. This provides excellent scalability as servers are added to a cluster, because no individual server is congested with work, and it forms the foundation of Matrix Server's robustness, because there is no master server that could be a single point of failure.

- **High performance:** Because each server accesses disk directly, Matrix Server can support block-based applications like SQL Server that demand low-latency, high-performance I/O.

- **Robust:** Matrix Server was designed from the ground up to handle mission-critical, data center workloads. In particular, the cluster file system is designed to recover automatically from server failure, without disrupting other servers in the cluster. One or more servers can fail, or indeed a series of servers can fail in quick succession, and other servers will be unaffected and continue to conduct processing. In conjunction with the high-availability engine (described in the next section), this allows rapid application recovery from server failure.

- **Native:** The Matrix Server cluster file system is a full file system that resides in the operating system as a direct peer of NTFS. In a typical deployment, NTFS is used for the C: boot drive, and the Matrix Server file system is used for application data drives D:, E: and so on that are shared across multiple servers. (It also supports mounting on junction points; for example, C:\SQL Data could be a Matrix Server file system while C: remains NTFS.) PolyServe developed this file system using "IFSKIT" interfaces licensed from Microsoft—the same interfaces NTFS uses to communicate with the Windows kernel. Its architecture is optimized to support data center applications running in a shared data environment with multiple active servers.

- **NTFS-compatible:** The Matrix Server cluster file system appears to server-based applications to function in the same way as NTFS. It uses the same ownership and authentication information, Access Control Lists, file locking and sharing APIs, file name formats, directory change notification mechanisms and I/O interfaces as NTFS. It supports the high-performance unbuffered I/O interfaces used by SQL Server and works with standard backup tools from leading vendors.

Matrix Server's cluster file system provides excellent performance and scalability not only for block-based applications like SQL Server, but also for file-based applications like file serving, Internet Information Server and streaming media. More information about these applications, as well as more details about the internal architecture of Matrix Server—such as the symmetric Distributed Lock Manager that the cluster file system uses internally to coordinate operations among servers—can be found in white papers available on the PolyServe website at http://www.hp.com/go/PolyServe.

High-Availability and application control engine

While the Matrix Server cluster file system is designed itself to be able to recover automatically from server failures, preserving full application availability requires attention to other software and hardware components as well. The Matrix Server high-availability engine provides this layer of coverage, with the ability to detect and respond to various kinds of hardware and software failures. It also provides the basis for administrative control of where applications run within the cluster.

Matrix Server has built-in **monitors for hardware and networking health**, using a heartbeat mechanism across one or more Ethernets. In the event of network switch failure, it can automatically transition cluster traffic among up to four Ethernet networks, according to an administrator-chosen priority order. If a server fails, Matrix Server can automatically restart each application that had been running on the failed server on another server in the cluster, chosen from an administrator-provided ranked list. (The top server on the list that is functional will be chosen.) If the server returns to the cluster and is healthy, Matrix Server can either move applications back to that server or leave them running where they are, subject to policy chosen by the administrator.

It also provides **software monitoring**, since software can fail even while hardware continues to operate. If a server is running multiple applications of which just one fails, Matrix Server can restart that application only while others continue to run without interruption.

Note

HP PolyServe Software for Microsoft SQL Server protects all databases from hardware or software failure—without requiring complicated storage configuration or extra hardware.

To enable remote clients to find applications on the cluster regardless of which server happens to be hosting the application at the time, the engine provides built-in support for the concept of **virtual hosts**. A virtual host consists of an IP address and one or more applications that can be moved as a group from one server to another. When the applications move, the IP address is moved as well. As a result, clients can connect to these IP addresses and receive service regardless of which physical server is involved.

While the basic operation of the high-availability engine resembles many other clustering products, Matrix Server's approach based on shared data allows high availability to be achieved in a **simpler, more easily configured and more reliable** way. As described briefly in the preceding section on availability problems in SQL sprawl, a typical traditional cluster demands close attention to storage configuration for each server pair. In particular, in the traditional approach, the failure of a server requires its storage to be taken over by the backup server. This transition entails an unusual sequence of configuration changes in the cluster and its attached storage, and it is the administrator's responsibility to ensure that all the hardware involved is configured properly for this exceptional circumstance. It can, in fact, be hard to determine whether this transition will succeed without actually triggering a test failover—and doing that can interrupt application availability.

By contrast, in the Matrix Server approach, backup servers have access to the necessary data to take over an application at all times; no exceptional storage reconfiguration is required at the time of failover. If there is any doubt about a server's ability to access the necessary files, reassurance can be had at any time simply by logging in and navigating an Explorer window into the appropriate folder.
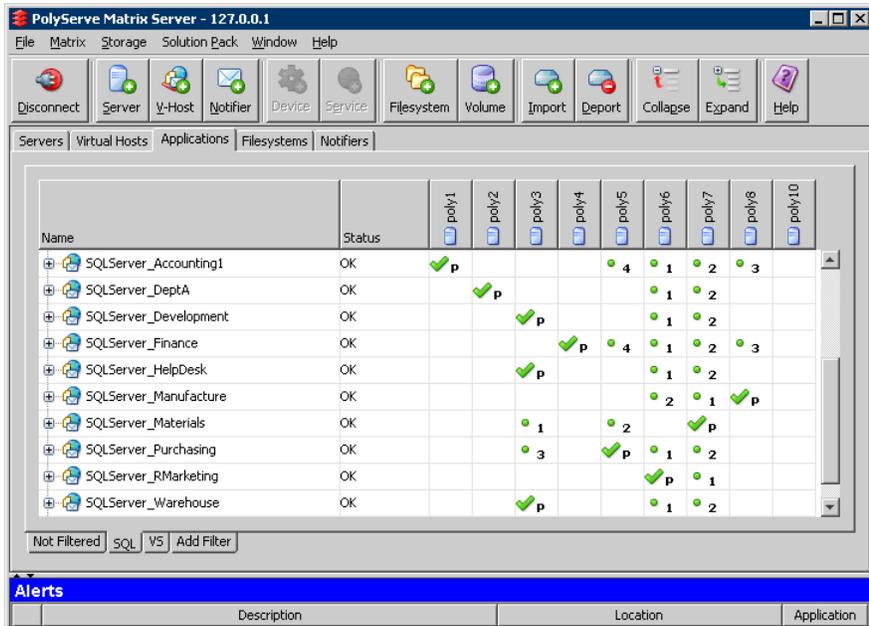
Finally, the high-availability engine, since it is able to start and stop applications anywhere in the cluster, provides the basis for **cluster-wide administrative control of applications and workload rebalancing**. To increase the utilization of a given server, an administrator can move a database from another machine in seconds. If a server is overloaded, the same process can shift work off it in seconds. Because of the shared data pool, no laborious after-hours migration of data is required. We call this capability to stop and restart database instances to shift workload Dynamic Rehosting.

Matrix Manager

HP PolyServe Software for Microsoft SQL Server is based on the principle of treating a collection of databases and the hardware resources that support them as a single unit. The foundation of this approach is the shared data pool enabled by the cluster file system, but its most concrete manifestation is Matrix Manager. Matrix Manager provides a single cluster-wide console that allows an administrator to see at a glance the status of all hardware and software components in the environment. It also provides a single point of control for configuring, updating, moving and protecting applications. See Figure 6.

Matrix Manager is an application that runs on an administrator's desktop or laptop and connects to the cluster over a secure TCP/IP session. After connected, the console displays all of the physical servers, storage, virtual hosts and applications in the cluster, with clear indications of health status. Figure 6, for example, shows a cluster with nine servers, indicated by columns. Ten database instances are visible in the display, although, as the scrollbar indicates, many more are actually present in the cluster. All ten visible instances are healthy, as shown by the "OK" status indication. The letter "P" indicates the primary server for each database, and the green check marks next to the "P" show that at the moment all databases are running happily on their primary servers. The circular bullets with the digits next to them show that the administrator has configured failover policies for each of these databases. For example, the Development, Helpdesk and Warehouse instances are currently running on their primary server, poly3, and are configured to fail over to poly6 or poly7, in order. The Accounting1 and Finance instances, on the other hand, have four backup servers configured.

Figure 6. Matrix Manager

Name	Status	poly1	poly2	poly3	poly4	poly5	poly6	poly7	poly8	poly10
SQLServer_Accounting1	OK	✓P				4	1	2	3	
SQLServer_DeptA	OK		✓P				1	2		
SQLServer_Development	OK			✓P			1	2		
SQLServer_Finance	OK				✓P	4	1	2	3	
SQLServer_HelpDesk	OK			✓P			1	2		
SQLServer_Manufacture	OK						2	1	✓P	
SQLServer_Materials	OK			1		2		✓P		
SQLServer_Purchasing	OK			3		✓P	1	2		
SQLServer_RMarketing	OK						✓P	1		
SQLServer_Warehouse	OK			✓P			1	2		

From the console, it is possible with just a few clicks of the mouse to:

- Add or remove servers from the cluster (while it is running, without incurring downtime)
- Add or remove storage from the cluster (while it is running, without incurring downtime)
- Create new virtual hosts or move a virtual host and associated applications to another server
- Specify a prioritized list of failover targets for a virtual host, should its primary server fail

- Examine a unified cluster log showing all significant events
- Create filtered views of the cluster status that highlight specific application or functional areas
- Implement notifiers that propagate extraordinary event indications, chosen by administrator-specified criteria, to pagers, email or trouble ticketing systems

For all these tasks, there is no need to connect to individual servers in the cluster to manage them. Just as the Matrix Server cluster file system is implemented symmetrically across all servers, so too does the management infrastructure operate symmetrically across all servers, with no need for a dedicated management server. Instead, all servers collaborate to maintain a consistent, replicated view of the overall cluster configuration. In fact, it is possible to reconfigure a server—what applications it should run, what drive letters it should use, what the failover policies should be for those applications, and so on—while it is down. When it recovers, it will heal itself back into the cluster, learn the current configuration and conform to it.

For those who prefer scripting, all the information available from Matrix Manager can be accessed by way of a command-line interface, and all the configuration tasks can be performed using command-line tools.
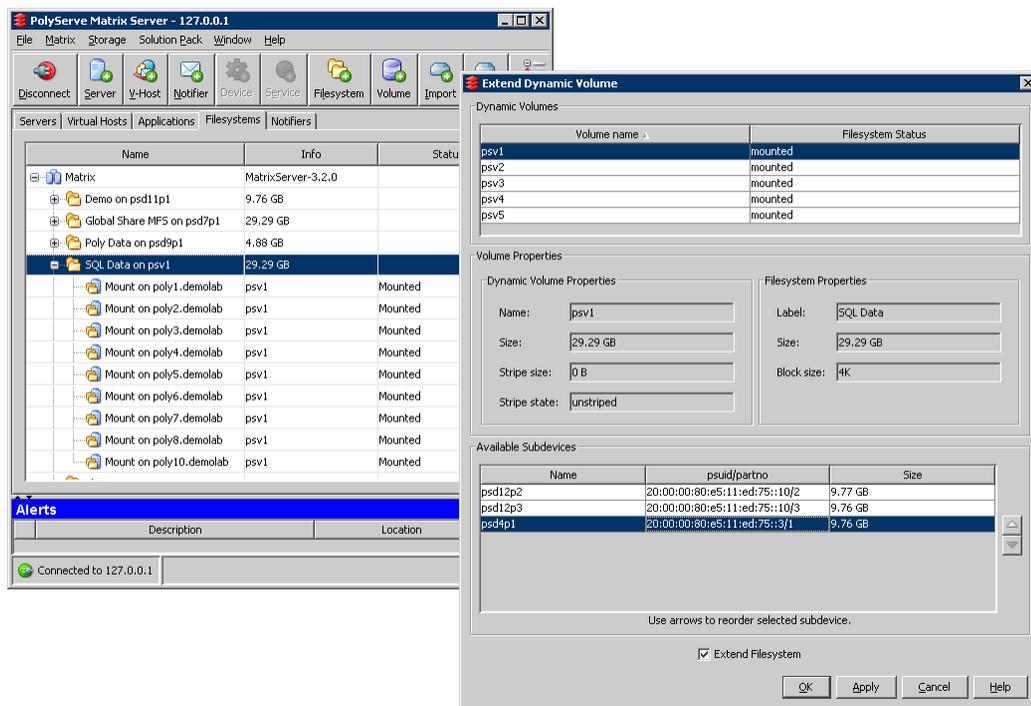
Volume Manager

Matrix Server provides the foundation for HP PolyServe Software for Microsoft SQL Server by allowing a set of servers to access shared file systems simultaneously and thus to be managed as a single unit. The Volume Manager provides an analogous capability for storage. It allows disk space from multiple storage arrays to be used and managed as a single pool.

- With the Volume Manager, an administrator can create a single volume from free space on multiple arrays (or in multiple LUNs on a single array). In this way a file system can make use of whatever storage is available.
- The Volume Manager also allows a volume to be striped across multiple arrays. This can improve I/O rates by aggregating the performance of the arrays, which is especially useful for sequential workloads frequently associated with data warehouses. To be precise, the Volume Manager allows a file system to be laid out across a concatenated collection of individual LUNs, stripe sets, or a mix of both. A stripe set is a set of LUNs of the same size across which Matrix Server will stripe data according to an administrator-chosen chunk size. For example, a stripe set with four 10GB LUNs and a stripe size of 64K will be treated as a single 40GB data container. The first 64K chunk of the container corresponds to the first 64K of the first LUN, the second 64K to the first 64K of the second LUN, and so on; then the fifth 64K chunk corresponds to the second 64K chunk of the first LUN, the sixth to the second chunk of the second LUN, etc. A contiguous interval of data is thus striped across all the LUNs in the set. When a volume is extended, an administrator can add either a single LUN (in which case the file system will grow into that space without striping), or a new stripe set consisting of multiple LUNs.
- Through concatenation or striping, Volume Manager permits the construction of huge file systems that exceed the typical 2-TB limit on the size of individual LUNs.
- In some environments, it is customary to configure every storage array with a set of LUNs of a fixed size—say, 30 GB. The Volume Manager allows a server administrator to construct file systems of whatever sizes are desired using these fixed LUNs.
- Finally, if a file system is becoming too full, the Volume Manager can be used to expand the file system, without taking the cluster down, using free space from any array accessible to the cluster.

Figure 7 shows how easily space can be added to a file system, even while that file system is in use cluster-wide. In this case, the administrator is adding roughly 10 GB of space to the "SQL Data" file system, which is mounted on all the servers in the cluster.

Figure 7. Adding space to a file system using the Volume Manager

Like all other aspects of HP PolyServe Software for Microsoft SQL Server, the Volume Manager is managed cluster-wide from a single control point. Thus, just as Matrix Server allows servers' data sets to be handled in a unified way, the Volume Manager allows the physical storage resources associated with a cluster to be used and managed as a single entity.

## HP PolyServe Software for Microsoft SQL Server

The final necessary ingredient is an integration layer that applies the core capabilities of Matrix Server and Matrix Volume Manager to SQL Server. This layer, called HP PolyServe Software for Microsoft SQL Server, includes:

- A SQL Server health monitor that periodically probes SQL Server instances within the cluster to ensure that client requests are being successfully handled. This will detect if a SQL Server instance hangs, even if the operating system and hardware remain healthy.
- A SQL instance virtualizer that allows the creation of a "Virtual SQL Server." A virtual SQL Server is an adaptation of the Matrix Server virtual host concept. It consists of a virtual IP address that clients use to connect, a specified primary server in the cluster, a prioritized list of backup servers, and from one to 16 associated SQL Server instances. If a server fails, the virtualizer will move the virtual IP address to the top server in the list that is capable of hosting the virtual SQL server; it then restarts the database instances in the new location. Note that, unlike server virtualization, the SQL instance virtualizer component itself is not active during normal operation. Thus there is no performance penalty; SQL runs at full native speed, with native access to all hardware resources. The virtualizer also ensures that a given database is only ever accessed from one server at a time, since SQL Server is not designed to allow multiple server concurrent access.

- A SQL Server registry replicator. SQL Server stores some configuration information in the Windows registry, which is located on each server's individual C: drive. To ensure this information is available to other servers in the cluster if a database instance needs to move, HP PolyServe Software for Microsoft SQL Server includes a component that automatically replicates relevant registry information into the cluster-wide shared storage.
- A SQL installation and hotfix updater agent. To simplify installation of SQL Server and of hotfixes across multiple servers, HP PolyServe Software for Microsoft SQL Server includes a push installer agent. The administrator simply places the appropriate installation packages in the shared file system, then uses the push installer agent to perform installations or hotfix updates across the entire cluster—what we call **"one-click maintenance."** Figure 8 shows this operation at work.

HP PolyServe Software for Microsoft SQL Server thus provides an easy way to move SQL Server instances within a cluster to improve resource utilization, an easy path to complete high availability for all instances in the cluster, and a simple way of performing typical SQL Server installation and maintenance tasks.

Figure 8. One-click maintenance allows SQL Server instances, service packs or hot fixes to be installed cluster-wide with a single click
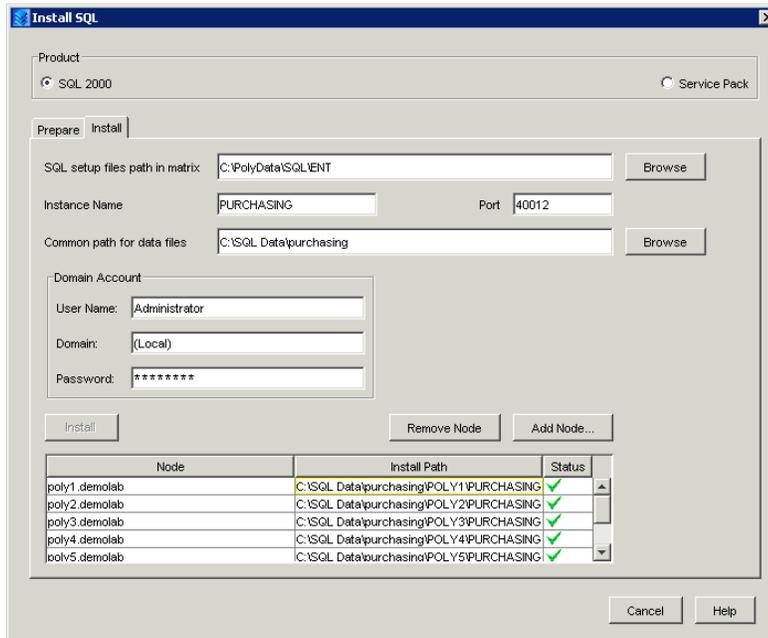
These capabilities are built on the core of Matrix Server and Volume Manager. The shared data pool provided by Matrix Server is used to store SQL databases and log files in a single location, accessible to all servers. The Volume Manager allows this storage pool to be spread across space on multiple storage arrays. Matrix Server's high-availability and application-control engine ensures that databases remain available regardless of server failures and, through Dynamic Rehosting, allows administrators to adjust work assignments to maximize utilization. Matrix Manager provides a single point for monitoring and controlling these elements.

The following sections explore in more detail how HP PolyServe Software for Microsoft SQL Server addresses the challenges of SQL sprawl.

Benefits of HP PolyServe Software for Microsoft SQL Server

Dynamic Rehosting: Consolidation without tears

The first and most obvious symptom of SQL sprawl is low utilization levels. The solution, of course, is consolidation, but in traditional approaches, SQL Server instance consolidation is a laborious and error-prone task. **The PolyServe Dynamic Rehosting capability makes consolidation a 10-second operation**; with a few clicks, database instances can be moved onto lightly loaded machines and the freed-up servers redeployed for other tasks.

Figure 9. Traditional database migration versus Dynamic Rehosting

Traditional Database Migration (6 hours) Starting Point: New server loaded with Windows		Migration with Dynamic Rehosting (51 mins) Starting Point: New server loaded with Windows	
21:00	**Document SQL parameters and directories on Old Server**	21:00	**Install Matrix Server and HP PolyServe Software for Microsoft SQL Server on New Server**
21:00	Make current backup of all databases	21:03	Configure server to join cluster
21:15	Shut down database on Old Server	21:07	Start Matrix Manager
21:20	Copy the SQL Server folder to New Server	21:08	Install SQL Server, service packs, and hotfixes on New Server with One-Click installation
21:22	Export SQL Server registry keys	21:38	Verify that New Server can see necessary database files in shared file system
21:25	Detach database from source instance Copy database to destination instance disk Attach database to destination instance disk	21:39	Use Dynamic Rehosting to move SQL Server instance to New Server
22:25	Move msdb database to New Server	22:41	Test SQL Server and connectivity to ensure successful move
22:45	Move master database to New Server	22:51	Complete
23:05	Move the model database to New Server		
23:25	Copy the registry keys to New Server		
23:30	Backup		
23:45	Install SQL Server on New Server (parameters and directory structures must be the same)		
00:45	Install service packs and hotfixes to New Server		
01:15	Stop SQL Server on New Server		

01:20	Export SQL Server registry keys
01:25	Copy the SQL Server folder on New Server
01:35	Copy SQL Data from Old Server to New Server over network
02:15	Shut down and power off Old Server
02:20	Import SQL registry keys on New Server
02:25	Change New Server IP address to Old Server IP address
02:30	Change New Server name to Old Server
02:35	Reboot New Server
02:45	Start SQL Server on New Server
03:00	Test SQL Server and connectivity to ensure successful move
03:10	Complete

Furthermore, everyone has heard stories of over-consolidation: after spending perhaps weeks planning a consolidation effort and a weekend implementing it, the administrator comes in to work Monday morning to discover unhappy users complaining their databases are no longer performing adequately. Or perhaps the initial consolidation goes well, but over time workload grows and performance suffers. In the traditional approach, partially backing out from a consolidation to offer more capacity to a given database is itself a painful, laborious task.

By comparison, with Matrix Server, if a user complains about poor performance, an administrator can start up Matrix Manager and, in a matter of seconds, shift the database onto a less heavily loaded machine. The user is happy and the pressure is off. (Of course, since the process is so quick, the administrator can shift the database back if desired, too!)

Dynamic Rehosting gives administrators the ability to match server capacity to workload need. Since heterogeneous clusters containing servers of different sizes are supported, databases that have periodic spikes in usage can be moved temporarily to a larger machine. For example, if a particular database is heavily used just at quarter-end, it might make sense to give it a dedicated 4-way server for those days, while at other times sharing a 2-way with other instances.

Easy, universal high availability

The second major problem posed by SQL sprawl is the fact that, despite using more hardware than necessary, it offers little actual redundancy to protect databases: high availability remains complicated to configure, expensive and, therefore, rare. HP PolyServe Software for Microsoft SQL Server, while reducing the hardware resources that are required, provides universal high availability to all the databases, and it does so in a way that is trivial to configure. Simply by using Matrix Manager to specify a list of backup servers for a given database, an administrator can ensure against the failure of any single server.

In the HP PolyServe approach, there is no need to worry about complicated storage configuration, SCSI RESERVE and RELEASE commands, and risks of failed failovers. In normal operation, it is easy to check that all servers have working access to the cluster file systems containing the databases. There is also no need to buy matched sets of hardware; since Matrix Server supports mixed clusters of servers from different vendors with different specifications, high availability can be achieved using hardware that is already in place.

Finally, HP PolyServe Software for Microsoft SQL Server offers **special support for failing over default (unnamed) instances**. In many traditional environments, there can be only one default instance in an entire cluster. This means high availability for default instances requires as many clusters as instances, with at least two servers in each cluster—or at least twice as many servers as default instances overall. PolyServe allows multiple default instances within a cluster (though of course only one instance can run on any one server at a time), and it also allows multiple default instances to fail over to the same target. For example, it is possible to have seven default instances in a cluster of eight servers; if any of the servers fail, the default instance from that server can fail over to the eighth server. In fact, a default instance can be given a list of failover servers; if another default instance has already failed over to the first server on the list, the database will choose the second one, and so on. Thus, for example, 14 default instances can run in a cluster of 16 servers and still provide complete high availability even if two servers fail. In this approach, default instances become highly available at the cost of only one extra server for every seven instances.

Maintenance off-loading

Of course, hardware and software failure is not the only source of downtime. In many environments, planned downtime for upgrades and patches is as great a concern. HP PolyServe Software for Microsoft SQL Server provides a solution for this, as well.

When it is necessary to perform a hardware or software upgrade or to apply a patch, databases are simply moved off the affected machine using Dynamic Rehosting. The machine can be taken off-line, upgraded, and then brought back into the cluster. Like server virtualization, this provides a way to perform hardware updates without long periods of application downtime. Unlike server virtualization (where migrating a database from one server to another takes the associated operating system instance with it), this also applies to operating system updates and patches.

Simplified management

Finally, the gravest problem presented by SQL Server sprawl is, as describedin the preceding section, the tremendous ongoing administrative burden it represents. Administrators who have to worry about monitoring and maintaining tens, hundreds or in some cases thousands of servers running SQL Server no longer have time to work on projects that can directly improve business results. HP PolyServe Software for Microsoft SQL Server can alleviate this problem several ways.

Figure 10. HP PolyServe Software for Microsoft SQL Server makes it easy to consolidate multiple instances on individual servers, raising utilization levels and freeing up resources for other uses
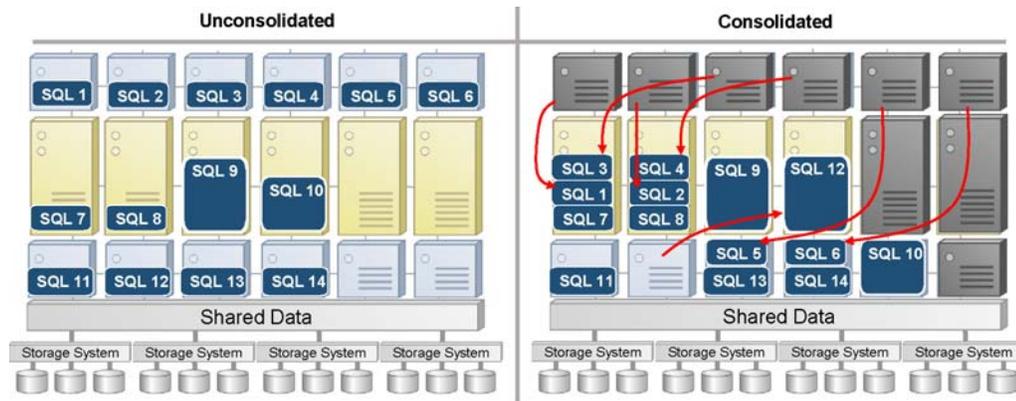
Most simply, by facilitating **consolidation onto fewer servers**, as shown in Figure 10, HP PolyServe Software for Microsoft SQL Server reduces the amount of administrator time consumed by the SQL infrastructure. And since administrators can easily flex capacity to meet changing needs—without running the risk of over-consolidating and creating performance problems—it is possible to maintain high levels of utilization as workloads change.

Note

> ...customers' results proved that they can achieve a dramatically more manageable and flexible data center—reducing server count and storage costs by as much as 50 percent.
> Microsoft Innovation
> Brochure

One-click maintenance can dramatically reduce the number of tasks associated with ongoing SQL Server operations. For example, one company started with a sprawling environment of over 110 databases running on the same number of servers. By using HP PolyServe Software for Microsoft SQL Server to consolidate onto three clusters of 16 servers each, the number of operating system images that needed to be maintained was reduced by over 50%. In fact, thanks to one-click maintenance, the customer was able to reduce the number of tasks involved in installing SQL hot fixes to just three: one for each cluster, a reduction of 97%.

In addition, Matrix Server simplifies administrators' lives by removing the complexity that has traditionally been associated with implementing high availability. With HP PolyServe Software for Microsoft SQL Server, all databases become highly available, without requiring many extra servers and without introducing complicated server or storage configuration tasks.

In fact, **storage management becomes dramatically simpler** with HP PolyServe Software for Microsoft SQL Server, as shown in Figure 11. The number of pools of storage that must be managed is dramatically reduced. Instead of having multiple drive letters for each database in an environment, as is often the case today, many databases can reside in a small number of drives that are shared across the entire cluster. There's no need to monitor free space on each server, just in the few file systems that are shared cluster-wide. In addition, backups can be simplified. By having SQL Server write hot backups to a drive letter shared across the entire cluster, it is possible to have just one tape backup job cover all instances. Indeed, there may be no need to install backup software on more than one server in the cluster.

With Matrix Manager, administrators have a dashboard that allows them to see all the databases and hardware resources in a cluster at a single glance, and have **a single point of control** that allows resources to be brought rapidly to need. HP PolyServe Software for Microsoft SQL Server makes even huge SQL Server environments manageable as never before.

Figure 11. Whereas stand-alone and traditional fail-over clusters entail complicated storage configurations, HP PolyServe Software for Microsoft SQL Server allows many servers to share a common storage infrastructure.
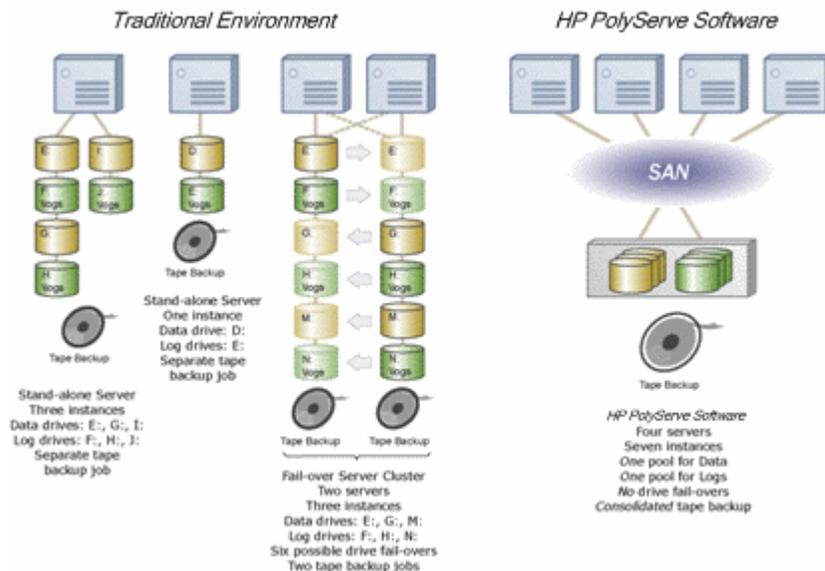
Supportability

Of course, all these attractive benefits would be for naught if the resulting environment were unsupportable. Before being acquired by HP, PolyServe was a Microsoft Gold Certified Partner, and had an exceptionally strong working relationship with Microsoft; this relationship continues today as a focus between HP and Microsoft. PolyServe worked closely with Microsoft to ensure supportability of the software, including licensing appropriate interfaces from Microsoft, such as the IFS Kit mentioned in the preceding section, to provide fully Windows-native and Windows-compatible functionality. In addition, PolyServe partnered with Microsoft Premier Support to ensure that Premier Support customers benefit from a tight interlock if any support questions do arise.

HP PolyServe Software for Microsoft SQL Server has passed Microsoft's rigorous review for inclusion in the SQL Server Always On program, and Microsoft support personnel are trained on PolyServe software products. In 2007, PolyServe was also recognized by Microsoft as Partner of the Year in the Advanced Infrastructure Solutions category.

An important part of what makes HP PolyServe Software for Microsoft SQL Server easy to implement and easy to support is its non-intrusive nature. It is delivered as a standard MSI installer file, and it runs with unmodified, regular versions of Windows Server and SQL Server which are, themselves, installed in the standard way. There's no need for extensive retraining of system or database administrators; anyone who knows how to manage SQL Server in a stand-alone environment can immediately be productive within a Matrix Server consolidation. The result is a significant reduction of support burdens.

Initial responses revisited

Having described how HP PolyServe Software for Microsoft SQL Server **shared data clustering** solution provides powerful benefits to SQL Server consolidation, it is perhaps useful to compare this approach to the traditional alternatives discussed in the preceding section.

Since the Matrix Server architecture uses a SAN, it retains the benefits that a SAN offers for storage provisioning. You can allocate more storage to a cluster as data footprint needs grow, without having to shut down the cluster or interrupt client access.

Like traditional consolidation on large servers, HP PolyServe Software for Microsoft SQL Server approach offers higher levels of utilization, but without necessarily requiring the purchase of new hardware. And, of course, in this approach all databases are given high-availability protection, and it is easy to rectify over-consolidation, if it happens, by shifting workload off overloaded machines.

Table 2. Server Virtualization compared with HP PolyServe Software for Microsoft SQL Server

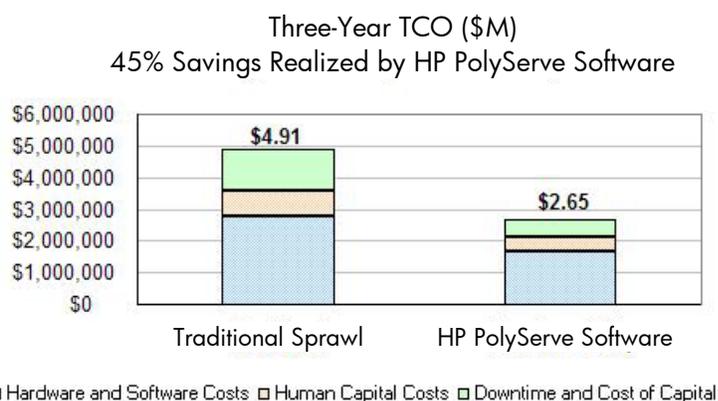| Server Virtualization…	HP PolyServe Software for Microsoft SQL Server…
Does not reduce the number of software images that need to be managed or simplify administration of SQL Server	*Does* reduce the number of software images, and *automates SQL Server with one-click cluster-wide installation and hot fix application*
Requires large, expensive servers	Uses *existing servers*
Imposes a performance overhead, by imposing a virtualization layer between the OS and the hardware	Runs SQL Server and the OS directly on the hardware, providing *full native performance*
Limits the hardware resources that can be given to a database	Imposes *no limits*, allowing any number of CPUs and gigabytes of RAM to be used for a database
Does not address high availability	*Make high availability easy* without requiring duplicate hardware or duplicate OS installations
Provides the ability to move a running database from one machine to another without restating, but not if high-availability software is being used.	Provides the ability to move a database from one server by stopping and restating it in seconds, and is *completely compatible with built-in high availability*
Raises supportability questions—Microsoft does not support Microsoft software running in conjunction with non-Microsoft hardware virtualization software	Is *completely supportable*
HP PolyServe Software for Microsoft SQL Server offer the best features of the traditional approaches without their drawbacks	

Total cost of ownership

Working with customers, HP has developed a simple Total Cost of Ownership (TCO) model that can be used to evaluate the potential savings offered by the HP PolyServe Software approach. For example, consider a traditional deploying supporting 18 databases on 20 servers, of which 16 are configured to run stand-alone and four are configured as active-passive failover pairs. With a conservative application of HP PolyServe Software for Microsoft SQL Server approach, the number of servers can be reduced in half, to a total of 10. The resulting savings is dramatic:

- Hardware and software capital expense can be cut from roughly $2 million to $1.2 million, with HP PolyServe software licenses included for HP PolyServe Software for Microsoft SQL Server approach. (Of course, if reusing already-purchased hardware, this is a measure of the amount of capacity which is freed up for other purposes.)
- Three-year maintenance and facilities costs can be cut from roughly $735,000 to $355,000 (including 24/7 support for the PolyServe software, in the second case)
- Operational costs cut from $814,000 to $479,000
- Cost of downtime cut from $576,000 to $72,000

These figures are based on conservative assumptions. For example, the operational costs use IDC's figure that 15-20 servers will consume 50% of the time of a system administrator costing $100,000 per year and a database administrator costing $130,000 per year. All operational savings derive from the reduction in the number of servers, with no assumption of savings from the reduced per-server effort (such as one-click maintenance) afforded by HP PolyServe Software for Microsoft SQL Server approach. The downtime costs are based on the assumption that entirely unprotected servers achieve greater than 99.995% reliability (roughly one minute of downtime per server per month), that HP PolyServe Software for Microsoft SQL Server reduces this a mere four-fold, and that downtime costs are slightly less than half the Gartner Group's estimate of $86,000 per hour for a large company. The overall results of the model, which includes other elements beyond those summarized here (such as cost of capital), are summarized in the following graph.

Figure 12. Three-year TCO

Three-Year TCO ($M)
45% Savings Realized by HP PolyServe Software

☐ Hardware and Software Costs ☐ Human Capital Costs ☐ Downtime and Cost of Capital

Conclusion

The success of SQL Server has given rise, inevitably, to a huge growth in the number of servers dedicated to hosting it. The result of this proliferation has too often been low levels of utilization, high capital costs, large amounts of stranded capacity, poor availability and—as the most dire consequence—an ongoing management burden.

Fortunately, there is a way to deploy SQL Server that can flex capacity to need, reduce capital costs, improve availability and dramatically simplify management. HP PolyServe Software for Microsoft SQL Server based on shared data clustering allows servers and storage to be brought together into a unified pool to satisfy SQL Server needs, offering the following benefits:

- **Dynamic Rehosting**, which allows databases to be moved among servers easily, maintaining high levels of utilization without risking overload
- **High availability** for all databases without the requirement to double up hardware
- **"One-click" maintenance** that allows administrators to update dozens of SQL Server instances in a single operation

In many real customer scenarios, HP PolyServe Software for Microsoft SQL Server has achieved dramatic improvements in capital effectiveness and ongoing operational expense:

- Server counts are typically reduced by 50% or more
- Configuration and maintenance time can be reduced by 75%
- Total Cost of Ownership over a three- or four-year deployment cycle can be reduced by as much as 70%

HP PolyServe Software for Microsoft SQL Server is the solution to SQL Server sprawl. Join us for an online demo to learn more about HP PolyServe Software.

For more information

- HP PolyServe Software best practices white papers (Disaster Recovery, Migration, Instance Aliasing, Sizing, Storage Deployment Guidelines, and Backups and Snapshots)
  http://www.hp.com/go/polyserve/bestpractices

- HP PolyServe Software for Microsoft SQL Server literature (overview, specifications, datasheet, solution papers, Q&A)
  http://www.hp.com/go/polyserve

- HP PolyServe Software storage solution for Microsoft SQL Server Always On program
  http://h71028.www7.hp.com/ERC/downloads/4AA1-5402ENW.pdf

- Application Platform Optimization for Microsoft SQL Servers
  http://h71028.www7.hp.com/ERC/downloads/4AA1-5401ENW.pdf

- Shortcut Guide to Microsoft SQL Server Infrastructure Optimization eBook
  http://www.hp.com/go/polyserve/bestpractices

- ESG Lab Report: HP PolyServe Software for Microsoft SQL Server—Highly Available SQL Server Consolidation
  http://h71028.www7.hp.com/ERC/downloads/4AA1-6316ENW.pdf

- HP StorageWorks SAN design reference guide
  http://h20000.www2.hp.com/bc/docs/support/SupportManual/c00403562/c00403562.pdf

- SQL Server 2005 Database Mirroring Integration with HP PolyServe Software
  http://h71028.www7.hp.com/ERC/downloads/4AA1-6313ENW.pdf

- Easing the Migration to Microsoft SQL Server 2005
  http://h71028.www7.hp.com/ERC/downloads/4AA1-5402ENW.pdf

- Microsoft SQL Server
  http://www.microsoft.com/sql/default.mspx

- HP StorageWorks for Microsoft Windows
  http://h18006.www1.hp.com/storage/osmswindows.html