

Scaling Out SQL Server

with Data-Dependent Routing

To scale out database applications efficiently and cost-effectively, enterprises can use a data-dependent routing (DDR) method to partition and access data across an array of industry-standard, symmetric multiprocessing nodes. This article explains how administrators can implement a DDR approach to scale out highly available Microsoft® SQL Server™ database-based enterprise applications, and then presents benchmark results from a pilot study in which Microsoft engineers simulated a real-world enterprise scenario—part of the Communication Services Platform for MSN (The Microsoft Network). The pilot configuration ran Microsoft SQL Server 2005 Beta 2 on Dell™ PowerEdge™ servers, Dell/EMC storage, and Emulex network adapters.

BY MAN XIONG, BRIAN GOLDSTEIN, AND CHRIS AUGER

Related Categories:

Pilot study

Database

Dell/EMC storage

Emulex

Microsoft SQL Server

Microsoft Windows

Performance

Visit www.dell.com/powersolutions for the complete category index.

The past decade has witnessed tremendous growth in enterprise applications on the Internet, and along with that, an explosion of data storage requirements. Today, many organizations must contend with the possibility that millions of online users will be using the Internet for common transactions such as shopping, storing e-mail messages, and viewing financial information. Database systems are at the heart of most enterprise applications, and Microsoft SQL Server is a leading database platform in large data centers—particularly those supporting online transactions.

A scalable database platform allows application designers to start small and grow a system as large as necessary. Traditionally, administrators have achieved scalability with symmetric multiprocessing (SMP) servers—using a scale-up approach by adding processors, memory, disks, and network adapters to a single server when required. As the

workload increases, however, a single database server—also referred to as a node in this article—will eventually hit a bottleneck and be unable to support further capacity. This situation typically becomes evident in terms of diminishing performance returns when additional system components are added to the chassis—or prohibitively expensive hardware upgrades to achieve the requisite performance.

To enable growth much beyond a factor of 10—which is generally considered to be the practical limit for scaling up a monolithic SMP server—application designers may adopt a scale-out architecture in which the workload and database can be partitioned among an array of industry-standard SMP nodes. A scale-out architecture enables administrators to grow systems incrementally by adding more nodes to the array. Such an array is often referred to as a federation of servers. Ideally, the partitioning is easy to manage and transparent to the application.

The scale-out approach based on a federation of servers is designed to provide several important advantages, including the following:

- **Cost-effectiveness:** SMP node arrays can be expanded in small increments of standards-based commodity components quickly and flexibly, in response to immediate business needs.
- **Fault tolerance:** Redundant system components can help prevent downtime so that a failure in one node does not necessarily cause the entire SMP array to become unavailable.
- **High availability:** The relative independence of nodes in the SMP array helps facilitate server failover, enabling high availability of enterprise applications.

Despite such attractive benefits, the scale-out approach can pose complex management challenges because a federation of servers typically involves many more components than a monolithic SMP server. Also, every application cannot be conveniently partitioned across nodes, so the scale-out approach may work well for some applications but be unsuitable for others.

Data-dependent routing

A key design decision for any scaled-out database platform is to determine the best way to partition data among the different nodes in an SMP array. Many applications can be partitioned by a particular value such as customer name, store location, time/date, register name, and so forth. The important consideration is how the data will be accessed. For example, the retail application for a large insurance company might be partitioned by branch office, allowing each branch office to maintain client records locally. In such a scenario, all data access would be local and a batch job could replicate new or modified records to the central SQL Server array at headquarters every night. In most cases, agents working at the branch offices would not access the central SQL Server array, and corporate analysts could run their reports against the central database without accessing every branch-office SQL Server node.

Data-dependent routing (DDR) requires intelligence in the client application, typically in the middle-tier layer, to route database requests to the appropriate node. However, the DDR approach does not provide views across nodes. With the exception of shared database schema, each node in a federation of servers is designed to operate independently. The middle tier contains the mappings to how data is partitioned and which node contains specific data.

In the preceding retail insurance scenario with partitioning, the application must be designed to track where records are located. This example assumes that a SQL Server database has been created on the middle-tier Web server. The database has been partitioned by customer ID across a number of federated servers, and a lookup table on the middle tier maps each customer ID to the node where the corresponding data partition resides.

Figure 1 shows a sample lookup table. When the customer service representative needs to display all transaction records for customer 10015, the application sends the request only to node 1. Because access is localized to a single node, no requests regarding customer 10015 are sent to nodes 2 and 3.

Since the application was partitioned by customer ID, records for each product ID could be stored on every database node. That would require the application to query every

Customer ID	Node ID
10015	1
10016	2
10017	1
10018	3

Figure 1. Lookup table mapping customer IDs to nodes where the corresponding data partitions reside

node, bring back all records that match each product ID, and then combine and sort the results. When product ID access is not localized, this could be a time-consuming operation. However, product ID updates could be run as a background batch job to help avoid slowing the response time for user transactions.

Challenges to the scale-out approach

When scaling out an application, administrators must address several complexities involving management, data partitioning, application development and revision, and high-availability practices. Such challenges include the following considerations:

- **Management:** The larger number of nodes in a federation of servers compared to a monolithic SMP server can increase overhead for operations management. For example, administrators must plan maintenance across an array of SMP nodes rather than a single node, and find ways to add and remove nodes without affecting application availability.
- **Data partitioning:** As enterprise applications grow, business needs may change—which may in turn require administrators to alter the way data is partitioned across the federation of servers. In addition, load balancing across an array of SMP nodes can be difficult to achieve because “hot spots” can develop on some nodes while other nodes remain relatively idle.
- **Application development and revision:** As business requirements and data access needs change, administrators must modify database schema—and find ways to implement these modifications without affecting application availability.
- **High-availability practices:** Administrators must determine how to handle single-node failures without disrupting application availability, and find ways to minimize the time it takes to restore a database to a single node.

Such challenges will be addressed in the following sections, which describe a successful scale-out scenario for a large-scale enterprise application.

MSN Communication Services Platform

At the heart of the popular Microsoft MSN® Messenger and Microsoft Hotmail® services is the Communication Services Platform (CSP) application, which manages contact information and buddy lists for these and other MSN applications. Voluminous contact information is stored in a large SQL Server database that is partitioned on 100 four-processor back-end servers running SQL Server 2000 Enterprise Edition. This federation of servers is designed to support millions of user accounts.

The CSP application provides an example of how SQL Server is designed to scale out using a federation of servers with DDR, helping to provide the following benefits:

- **Processing power:** The workload volume that can be handled by an extensible array of industry-standard SMP nodes has the potential to far exceed the processing power of any single-server SMP system.
- **Query isolation:** The isolation of queries on a per-user basis can make the MSN CSP application an excellent fit for row-based partitioning and DDR.
- **Intelligent partitioning:** Because the SQL Server database is designed for flexible data partitioning, the CSP application can be run cost-effectively on commodity hardware such as industry-standard, four-processor servers.

Figure 2 presents an overview of the system architecture that supports the Microsoft MSN CSP application. System components are configured in four tiers, as follows:

- Web servers running Microsoft Internet Information Services (IIS) 6.0
- Lookup partition database servers running SQL Server 2000 Enterprise Edition
- Database back-end servers running SQL Server 2000 Enterprise Edition
- MSN scale-out management layer server

Records are ordered and partitioned by Passport User ID (PUID) across the back-end database servers. The scale-out management layer stores the mapping of data partition IDs to the back-end database servers in its own configuration database. The lookup partition server database stores the mapping of PUIDs to data partitions. Clients of the CSP post requests to the Web server, which queries the lookup partition server with the PUID to obtain the data partition ID where the records are located. Then the Web server queries the scale-out management layer to determine which back-end database server contains the information for that user. Information is typically returned to the client in a matter of seconds.

MSN scale-out management layer

The MSN scale-out management layer provides a platform for the MSN CSP application to deploy the partitioning, DDR, and failover topology for the back-end database servers. The MSN scale-out management layer defines a *fail-safe set* as a set of databases that contains one primary database and its replicas, called *secondary databases*. A fail-safe set is designed to be the high-availability unit for the MSN scale-out management layer. In practice, the primary and secondary databases of each fail-safe set are placed on different servers to help ensure high availability. Primary databases and secondary databases are synchronized by SQL Server transactional replication for the CSP application. A *partition* is defined as a set of partitioned data, which is the unit for data partitioning and DDR. A partition is stored in a fail-safe set with its master copy on the primary database and its replicas on the secondary databases. A *failover group* is defined as a group of servers that function as backups for one another. Since workload goes only to primary databases for the CSP application, primary databases for partitions are placed across servers carefully to distribute and balance the workload among servers of the failover group. Failover groups are made independent from one another by not allowing a fail-safe set to go across a failover group boundary.

The example shown in Figure 3 is a simple failover group consisting of two servers. This group hosts two fail-safe sets, highlighted in two different colors. In this particular example, each fail-safe set stores only one partition and has only one secondary database. The data on the primary database is replicated to the secondary database. The primary database of fail-safe set 1 is placed on server 1, and its secondary database is placed on server 2. The primary database of fail-safe set 2 is placed on server 2, and its secondary database is placed on server 1. Both arrangements are designed for high availability. The primary database for partition 1 is placed on server 1 and the primary database for partition 2 is placed on server 2 to help balance the workload.

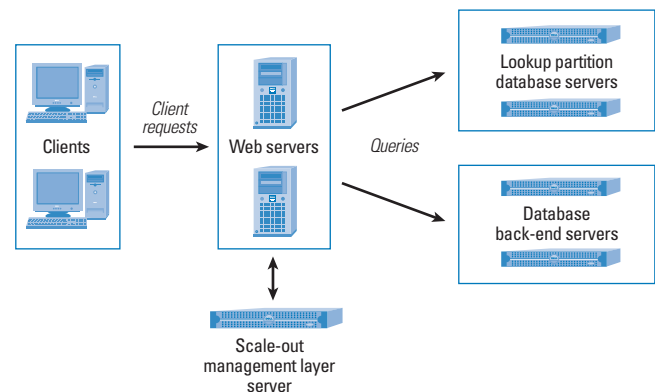


Figure 2. Overview of system architecture for the Microsoft MSN CSP application

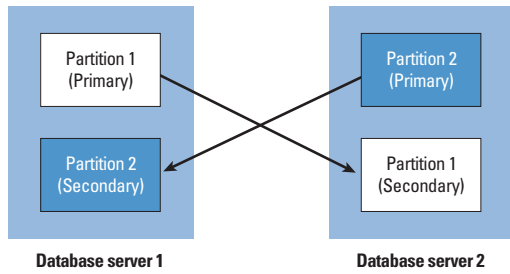


Figure 3. Replication-based fail-safe sets in a failover group

The scale-out management layer maintains a configuration database to store information about deployment and status, including:

- **Mapping:** The mapping of a partition to the SQL Server instance name and the database name of its primary database, by partition ID
- **Topology:** The topology of the fail-safe sets and failover groups
- **Status:** The current state of databases and SQL Server servers

The scale-out management layer reads configuration files for the partitioning and failover topology of the database servers and stores the information in its own SQL Server database, then configures the servers accordingly. It monitors the status of servers for failover operation and maintains the partition mapping for DDR. The scale-out management layer application also provides an administrative interface for common system maintenance operations on scale-out systems using replication to enable high availability, as shown in Figure 4.

Scalability to accommodate data and workload growth

The MSN CSP application architecture is designed to accommodate natural data growth and increased client requests. When address books are added to the system, the number of client requests increases and the CPU usage on the database servers increases. Best practices recommend a maximum operating usage of 50 percent across all CPUs to allow for the failover design shown in Figure 3. When the 50 percent CPU usage threshold is exceeded, administrators should add database servers to the system configuration. This is accomplished by adding a new failover group.

Additional failover groups

A new failover group can be added to the system by using the scale-out management layer interface. Administrators update the information in the scale-out management layer configuration database for DDR and failover topology. Lookup partition servers examine data distribution across the back-end database servers when new accounts are requested, so adding a new failover group results in new accounts being directed automatically to the new group. Figure 5 shows the

DDR and high-availability architecture of a system, before and after adding a failover group. Details of actual execution will be explained in the “Pilot study of the scale-out approach” section of this article.

Workload balancing

When a new account is created, lookup partition servers estimate how busy each database server is and add the new account to the least stressed server. This task is accomplished by determining a heuristic indicator of each database server’s load, which is calculated from the sum of existing accounts, weighted by the number of contacts in each account. The load of a database server is approximately proportional to the value of this heuristic indicator. Newly added database servers are configured to process new user accounts and the corresponding workload until their load reaches a level similar to that of the preexisting database servers. This approach helps ensure a smooth scale-out process that can enable the total throughput to scale linearly with the number of nodes.

Enhanced availability with transactional replication

The database uptime requirement for the MSN CSP application reads is 100 percent. Given the current Internet usage environment, 10 minutes of downtime per year is allowed for write access. The CSP application has enabled Microsoft to achieve these service levels in the two years it has been in operation.

Scale-out management layer administrative operation	Description
Promoting a database	Converts a secondary database into a primary database by redirecting workload and establishing replication from the primary to the secondary database.
Demoting a database	Converts a primary database into a secondary database. This results in draining the replication queue and dropping replication for that database. If this is the primary database in a fail-safe set, the appropriate secondary database will be promoted.
Marking a database as “offline”	Prevents client applications from querying a database and pauses all replication processes. If this is the only primary database, the appropriate secondary database will be promoted.
Marking a database as “online”	Resumes replication processes to and from a database.
Marking a database as “needs repair”	Results in draining the replication queue and dropping replication for a database. If this is a primary database, the appropriate secondary database will be promoted.
Repairing a database	Re-creates a database that is marked for repair through a backup/restore process, after which the database is left in an offline state.
Marking a server as “offline”	Causes all databases on a server to be marked as “offline.”
Marking a server as “online”	Causes all databases on a server to be marked as “online.”

Figure 4. Common scale-out management layer administrative operations

The MSN CSP application uses SQL Server transactional replication to help ensure high availability because transactional replication enables a combination of low latency and transactional consistency guarantees. The CSP application is not designed to read or write to both the primary and the secondary databases concurrently for two reasons:

- Application design would be much more complex. It would be necessary to implement bidirectional replication between primary and secondary databases.
- The secondary copies coexist with primary copies of different data set partitions on the same nodes. Reading from the secondary database would take resources from those primary databases.

Transactional replication uses the transaction log to capture incremental changes that were made to data in a published table. Microsoft SQL Server monitors INSERT, UPDATE, and DELETE statements, or other modifications made to the data, and stores those changes in the distribution database, which is designed to act as a reliable queue. Changes are then propagated to subscribers and applied in the same order as they occurred by opening a connection to the subscriber database and issuing SQL commands to the subscriber database. In applications where there are high write-to-read transaction ratios, replication may lag behind the transaction processing. The common limiting factors are network latency, index overhead on the subscriber database, and the number of connections to the subscriber that is executing the commands. When the source system’s transactions per second exceed the replication capacity of the subscriber’s system, replication latency will keep climbing until the transaction load is reduced. Losing the primary system causes transactions to be lost in the replication queue. The tolerable level of transaction loss depends on business needs and the desired end-user experience.

There are several ways to avoid the replication bottleneck:

- The data and workload should be spread out across more physical servers, which reduces the workload per distributor—although this approach can lead to under-utilized hardware. This is the option elected for the CSP application.
- Since each server running SQL Server has only one distributor, having multiple SQL Server instances provides multiple distributors per server. By spreading the original workload across these instances, the replication load can be processed by more distributors per server. In this way, additional servers are not needed but additional management is required for allocating hardware resources among instances, such as CPU and memory.
- Microsoft SQL Server 2005 is designed to support parallel replication and to help ensure that transactions are processed to the subscriber in the same order as to the publisher. At press time, SQL Server 2005 had not been released so this feature was not yet implemented on the CSP production site.

On an ongoing basis, the MSN CSP application development team runs a stress test lab at Microsoft to determine the stress-level threshold where replication queue build-up exceeds specified design levels. In live production, the MSN CSP operation team monitors the stress level of client requests per node. As the number of accounts and size of the accounts increase, so does the number of queries per node. When the stress level reaches the threshold, the CSP team adds another failover group to the system.

System failure detection and failover

The MSN scale-out management layer monitors the status of all nodes. It detects the failure of a server or a database and promotes the secondary database for the failed partition by redirecting the workload traffic to the secondary database.

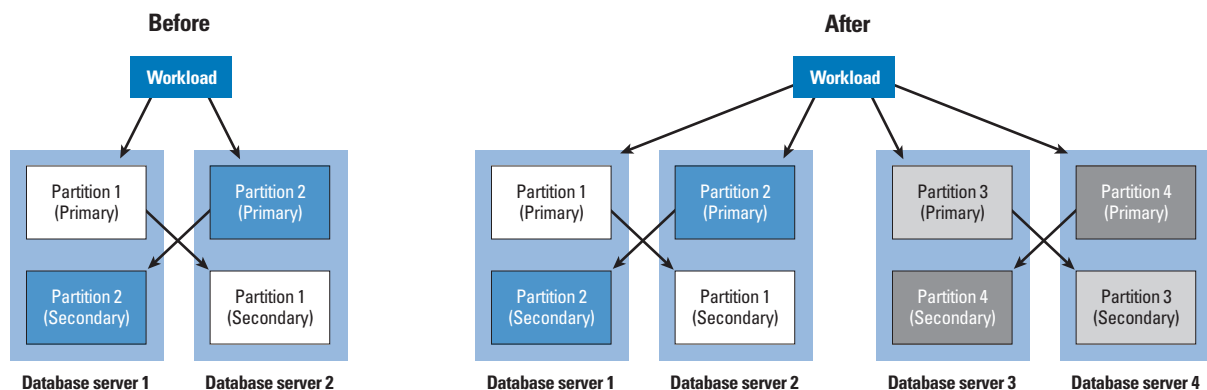


Figure 5. Adding a failover group

System role (number of systems)	Model	CPU	Physical memory	Storage	OS version	Application
Database server (4)	Dell PowerEdge 6650	4 Intel® Xeon™ processors at 2 GHz	8 GB	Dell/EMC CX600 array-based SAN	Windows Server 2003, Enterprise Edition	SQL Server 2005 Beta 2
Lookup partition server (2)	Dell PowerEdge 6650	4 Intel Xeon processors at 2 GHz	8 GB	Direct attach SCSI disk array with five 146 GB drives	Windows Server 2003, Enterprise Edition	SQL Server 2005 Beta 2
Web server (3)	Dell PowerEdge 2650	2 Intel Xeon processors at 2.4 GHz	4 GB	Local disk	Windows Server 2003, Enterprise Edition	IIS 6.0
Scale-out management layer server	Dell PowerEdge 2650	2 Intel Xeon processors at 2.4 GHz	4 GB	Local disk	Windows Server 2003, Enterprise Edition	
Web client (12)	Dell PowerEdge 1650	2 Intel Pentium® III processors at 1.4 GHz	2 GB	Local disk	Windows Server 2003, Standard Edition	

Figure 6. Configuration for pilot study deployment of Microsoft CSP running on SQL Server 2005 Beta 2

The Web server establishes connections to the back-end databases according to the information in the scale-out management layer configuration database. The Web server makes requests against the correct physical database instance during processing. Return codes from SQL Server indicate connection problems. Connection time-out is also treated as a failure. The Web server runs a client of the MSN scale-out management layer, which communicates the failure to the management layer. The management layer “blacklists” the failed databases and is designed to redirect the Web server to its backups in a matter of seconds.

Systems maintenance

Maintenance presents a particular challenge for scale-out systems. For example, an online transaction processing (OLTP) application like the MSN CSP application requires routine systems maintenance tasks including OS and application patching, node replacements and additions, database backups, and index defragmentation. Such common tasks must be accomplished in a way that incurs no appreciable impact on data availability and workload performance. Some maintenance tasks can be performed without taking databases offline; others require taking a particular database offline, while still others require bringing down an entire server. Administrators can use the administrative interface of the scale-out management layer for common system maintenance tasks.

Pilot study of the scale-out approach

In January 2005, Microsoft engineers configured an example enterprise deployment of the large-scale MSN CSP application in the Microsoft SQL Server Product Group Scalability Lab. This pilot study was designed to demonstrate how a scaled-out, end-to-end OLTP database application that enables high availability and enhanced manageability can be built using the Microsoft SQL Server 2005 platform.¹

The test configuration included 12 Web clients, three Web servers, two lookup partition servers, four database servers, and one scale-out

management layer server (see Figure 6). The three Web servers were connected to the network through a switch that distributes queries from clients to the Web servers in a round-robin pattern to enable load balancing. Data and log files for the database servers were stored across the same group of disks on a storage area network (SAN), which consisted of a Dell/EMC CX600 storage array with 146 GB 10,000 rpm drives. Each database server was connected to the 2 Gbps switched Fibre Channel SAN with two Emulex LP9802 Peripheral Component Interconnect Extended (PCI-X)-based host bus adapters to enhance I/O capacity and I/O failover.

The test team configured the enterprise scenario described in this article using SAN storage instead of direct attach storage (DAS) because the SAN approach is designed to enable the following benefits compared to DAS:

- **Centralized management:** Helps IT organizations streamline administration.
- **Flexible scale-out:** Enables administrators to increase capacity and/or the number of spindles without adding servers.
- **High availability:** Allows administrators to upgrade and deploy storage resources without disrupting business operations running on individual servers.

Although the SAN approach can be significantly more expensive to deploy and maintain than DAS, SANs can be used to enable sophisticated high-availability data deployments. However, that discussion is beyond the scope of this article. *Note:* Best practices recommend careful planning when using a SAN to support high data availability, to help avoid configurations in which a SAN may become the single point of failure for an entire enterprise system.

Increasing throughput with additional nodes

Figure 7 shows that the total number of transactions per second processed by the four database servers increased proportionally to

¹ The Microsoft SQL Server 2005 Beta 2 release was used in the pilot study described in this article. Actual features of Microsoft SQL Server 2005 as it is released to market may differ slightly; actual performance depends on various factors, including but not limited to the specific hardware and software configurations on which Microsoft SQL Server 2005 is deployed.

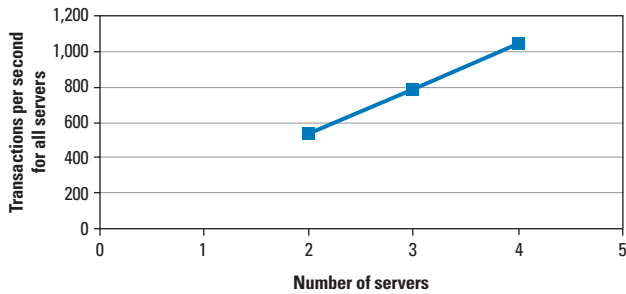


Figure 7. Workload performance scaling versus number of database servers

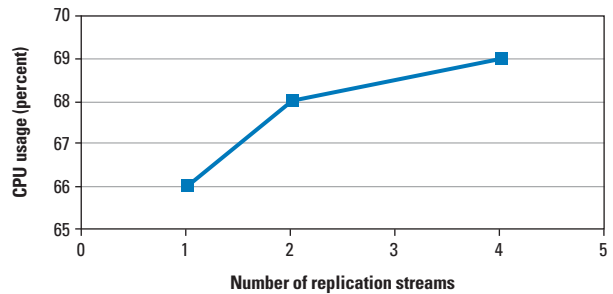


Figure 9. CPU usage versus number of replication streams

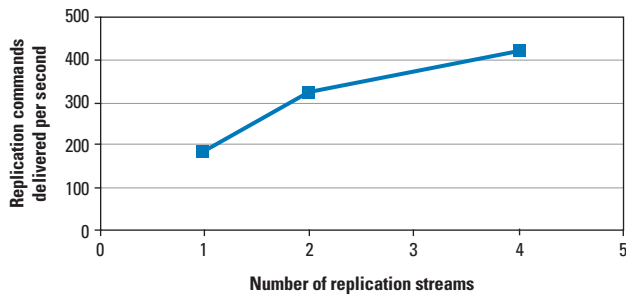


Figure 8. Replication throughput versus number of replication streams

- **Number of replication commands:** Run only the necessary number of streams; avoid adding streams if the replication queue can be drained in a timely way.

Figures 8 and 9 show that, in the pilot study, the replication throughput indicator (number of replication commands delivered per second) increased significantly when the number of replication streams was increased up to four streams. Each additional stream added roughly 1 to 2 percent of CPU usage.


the number of database servers in this pilot study. These results demonstrate that true linear scaling is possible with proper application architecture. The MSN CSP application is designed to be scaled out for any number of users; currently, millions of users are supported by the online MSN CSP application.

Combining the scale-out management layer and replication for enhanced availability

Together with the scale-out management layer application, replication can help provide excellent data availability. Single-threaded transactional replication in SQL Server 2000 prevented the MSN CSP application from fully utilizing available server resources. To help overcome that limitation, SQL Server 2005 is designed to support parallel streaming of replication commands—enabling multiple streams of 1 to 64 commands to be processed simultaneously. The optimal number of streams and how well the performance will scale typically depend on a variety of factors. Best practices include:

- **Number of CPUs:** Do not use more replication streams than the number of CPUs processing those streams to help minimize context-switching overhead.
- **Blocking:** Avoid overlapping transactions on tables being replicated to help prevent replication streams from blocking one another.
- **CPU cycles:** Be aware that adding a replication stream creates a task, thus requiring more work and potentially increasing CPU usage.

A highly manageable SQL Server scale-out system

In the example enterprise deployment of the Microsoft MSN CSP application, engineers observed linear performance scaling for up to four database servers. These results were achieved by partitioning and accessing data across an array of industry-standard SMP nodes that were configured as a federation of Dell PowerEdge servers. In addition, the pilot study explored ways to enable high availability using the scale-out management layer application in conjunction with SQL Server transactional replication. The findings discussed in this pilot study demonstrate that Microsoft SQL Server 2005 can be used to effectively scale out an enterprise database application using DDR to help accommodate data and workload growth. 

Man Xiong is a software design engineer in test on the Microsoft SQL Server product team. Man has a master's degree in Computer Science and Engineering from the University of Washington.

Brian Goldstein is a lead program manager on the Microsoft SQL Server product team. Brian has a master's degree in Engineering from the Massachusetts Institute of Technology and is a Microsoft Certified Systems Engineer.

Chris Auger is an enterprise technologist in the Dell Advanced Systems Group.

FOR MORE INFORMATION

Microsoft SQL Server 2005:
www.microsoft.com/sql/2005/default.asp