Global Knowledge™

Expert Reference Series of White Papers

Inside SQL Server 2008: Exploring High Availability and Scalability Enhancements

Inside SQL Server 2008: Exploring High Availability and Scalability Enhancements

Michael Manning, Global Knowledge Instructor

Introduction

Databases are at the heart of every organization. They support line-of-business applications, internet applications, human resource systems, financial applications, and more. Application downtime and database errors mean lost customers in both the short- and long-term. According to a 2007 University of Warwick study, hourly downtime costs were pegged at $682,000 per hour in the retail sector and $192,000 in the financial sector. Those costs are probably conservative and could be higher or lower, depending on the size and nature of your business. What are the causes? Causes include such factors as natural disasters, hardware failure, upgrades, human error, and others. Achieving high availability is a critical component of our job as database administrators (DBAs). Your final high availability (HA solution) will depend on what you are trying to protect against, what is your acceptable downtime tolerance, and what will operate along side your Disaster Recovery strategies.

SQL 2005 introduced high availability and scalability enhancements, such as database mirroring and improved failover clustering from earlier versions. SQL Server 2008 has built on the earlier SQL 2005 architecture and added some interesting new features. Microsoft is targeting close to zero downtime with SQL 2008 with availability and recovery features that only used to be available via third-party solutions. These features include prioritization management, hot-swap replication, and the ability to recover off of a mirrored server. This white paper discusses the evolution of High-availability from SQL 2005 to SQL 2008, and the tools available to minimize downtime in the Enterprise.

High Availability Options

SQL Server 2008 offers a variety of high-availability options to help minimize downtime. These features include:

- Database Mirroring
- Log Shipping
- Failover Clustering
- Peer to Peer Replication
- Database Snapshots
- Dynamic Configuration

Let's take a look at these options and how they can positively affect high-availability and disaster recovery needs.

Database Mirroring

Database mirroring offers increased database availability by maintaining a hot standby database on another instance of SQL Server 2008. The mirror database is an exact copy of the principal database, and, as changes are made to the principal database, they are automatically synchronized to the mirror. Database mirroring works by transferring and applying a stream of database log records from a principal to a mirror database. In case of a failure on the principal server, client applications are automatically redirected to the mirror server without any changes to the application. The standby database becomes the active database, and clients are redirected without any data loss or downtime to the organization. Database mirroring can be configured using the Database Mirroring Wizard (see Figure 1).

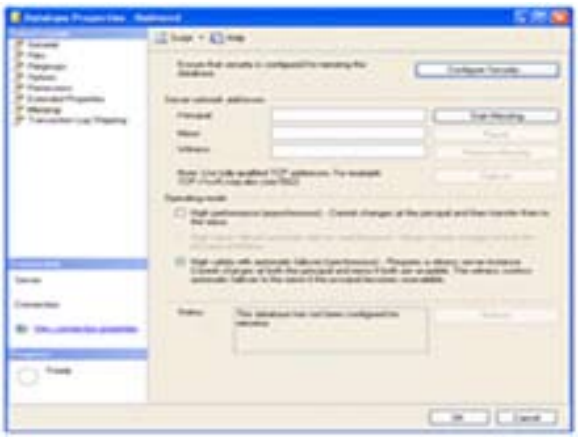

Figure 1: Database Mirroring Wizard

Database mirroring fills two roles, firstly as an HA solution for a local site and secondly as a vehicle to meet disaster recovery requirements. As part of an organizational disaster recovery plan, a hot or warm standby database could be placed in a separate geographical location from the primary active database. As an example, the principal database could reside in the Seattle data center, and the mirror database in the San Jose disaster recovery site (see Figure 1). The principal database in the Seattle data center handles all client operations while the mirror database receives a steady stream of transaction log changes through a dedicated TCP endpoint. This ongoing process keeps the mirror database synchronized with the active database and ready to take on client activities in case of a failure.

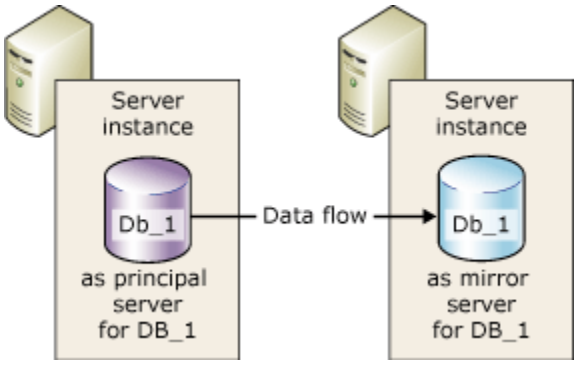

Figure 2: Seattle Principal and San Jose Mirror Servers

Database mirroring can be configured with one of two operating modes: high-safety and high-performance.

High-safety mode supports synchronous operations and applies changes to the principal and mirror servers in unison. As soon as the principal and mirror databases are synchronized, then the transaction is committed to both partners in real-time. This two-way commit has a small cost in terms of performance, but ensures that data will not be lost between the principal and mirror databases. Automatic failover between principal and mirror in High-safety mode requires a third partner known as the witness server. The witness server is constantly checking to make sure that the principal is up and running. If the mirror and the witness lose the connection with the principal, then the mirror server will automatically failover.

High-performance mode supports asynchronous operations so the principal server will commit transactions without waiting for an acknowledgement from the mirror server. The mirror database could lag behind the principal, depending on the work loads of the two partners. Running operations asynchronously enables the principal to run with very little transaction latency, but at the potential cost of some data loss. Failover is manual if the mirror is set to High-performance mode.

Mirroring Enhancements in SQL 2008

Automatic recovery from corrupted pages. SQL Server 2008 Enterprise edition introduces mirror protection down to the data page level that can asynchronously pull pages between the principal and mirror databases to fix corruption. A database mirroring partner automatically tries to resolve certain types of errors (823, 824, and 829 errors). If a page is corrupt in the principal database, then a query hitting the page will fail. Once the page is discovered, it should generate an 829 error until it is fixed. If a page is corrupt on the mirror, then the mirroring session will be suspended until the page is fixed. Once all of the corrupt pages have been fixed the mirroring session will resume automatically. Pages are repaired asynchronously and, as long as the page is not corrupt on both the mirror and principal, the matching non-corrupt page will be recovered from the partner server, thereby resolving the error, and database operations continue without a glitch. There is a new DMV – **sys.dm_db_mirroring_auto_page_repair** – that will allow you to track all corrupt pages in a mirrored database. All in all, this is a great new feature in SQL 2008, but this does not mean that you can ignore the various errors when they occur. It is still important to do a root cause analysis so that corrupt pages can be prevented before they occur.

Automatic page recovery is a feature that is only available in SQL 2008 Enterprise edition. If the principal is running Enterprise edition and the mirror is running Standard edition, then corrupt pages can be repaired from the mirror but not the other way around.

Database Mirroring Log Stream Compression. An additional feature was introduced in SQL Server 2008 called "Database Mirroring Log Compression." Mirroring works by shipping transaction logs from the principal to the mirror database and then applying those transactions to the mirror. Increased network traffic can be a concern, as these transaction logs are sent across the network, and this is even a greater concern on mirrored databases running in High-safety mode. Log compression is a way to ease this problem. Microsoft claims a minimum compression rate of 12.5%, depending on the data that is being processed by the application (SQL Server 2008 Books Online). There is a cost to this compression, however, and that comes in the form of the extra processing power required to compress the transaction logs on the principal and then decompress them

on the mirror server. This extra CPU load could be a problem for some systems that are already operating at the upper range of their processing capabilities; they could actually see a drop in performance if log compression is enabled. Log compression is on by default in SQL 2008, but this feature can be disabled.

Database mirroring in SQL Server 2008 offers high availability and provides a manageable alternative or complement to failover clustering or log shipping. Database mirroring is supported in SQL Server 2008 Standard and Enterprise editions.

Log Shipping

Similar to database mirroring, log shipping is another SQL 2008 HA solution offering increased database protection by maintaining a warm standby database on another SQL Server instance. Log shipping allows you to take automated transaction log backups from a primary server and send those backups to one or many secondary servers to be restored. Unlike database mirroring, which allows one principal and one mirror only, log shipping can utilize several warm standby databases which can increase the level of protection. Log shipping failover, however, is not automatic; several steps must be performed to complete failover in the event of primary database failure. Applications using the primary database must be manually redirected to the secondary database after bringing that database online. Log shipping can also use backup compression to reduce the size of log files to be sent over the network.

Log shipping is set to run on a schedule, and there will be a delay between data changes on the primary and standby servers. Log shipping is supported on both the Enterprise and Standard editions of SQL 2008.

Log Shipping Enhancements in SQL 2008

Sub-Minute Log Shipping in SQL 2008. SQL Server 2005 Management Studio allowed the frequency of scheduled jobs to equal 1 minute or more, meaning that the minimum latency for a log shipping solution could be as long as 3 minutes (1 minute each for backup, copy, and restore jobs). One of the great new features in SQL Server 2008 is sub-minute log shipping jobs. In fact, SQL 2008 allows log shipping jobs to be scheduled with the frequency in the seconds, minutes, hours range using the Management Studio and stored procedures (see Figure 3 below).

There are considerations that you should be aware of when setting up frequent log shipping jobs. The next scheduled execution of the job will not occur until the previous execution has completed and could, in fact, be skipped if it goes beyond the scheduled frequency interval. So, if the frequency interval of jobs is set to 1 minute, but the actual job takes to 1-1/2 minutes to complete, then the job scheduled to start at the 1 minute mark would be skipped, and the next job would begin at the 2 minute mark. One execution of the job would be skipped in this scenario.

Information about each backup job is recorded in the log files as well as the msdb database, in the dbo.backupset, dbo.backupmediaset, dbo.backupmediafamily, dbo.backupfile, and dbo.backupfilegroup tables. If you back up frequently, these tables will grow quickly, so it is a good idea to check the size of these tables from time to time and delete or archive as necessary, using the delete backup history stored procedure.

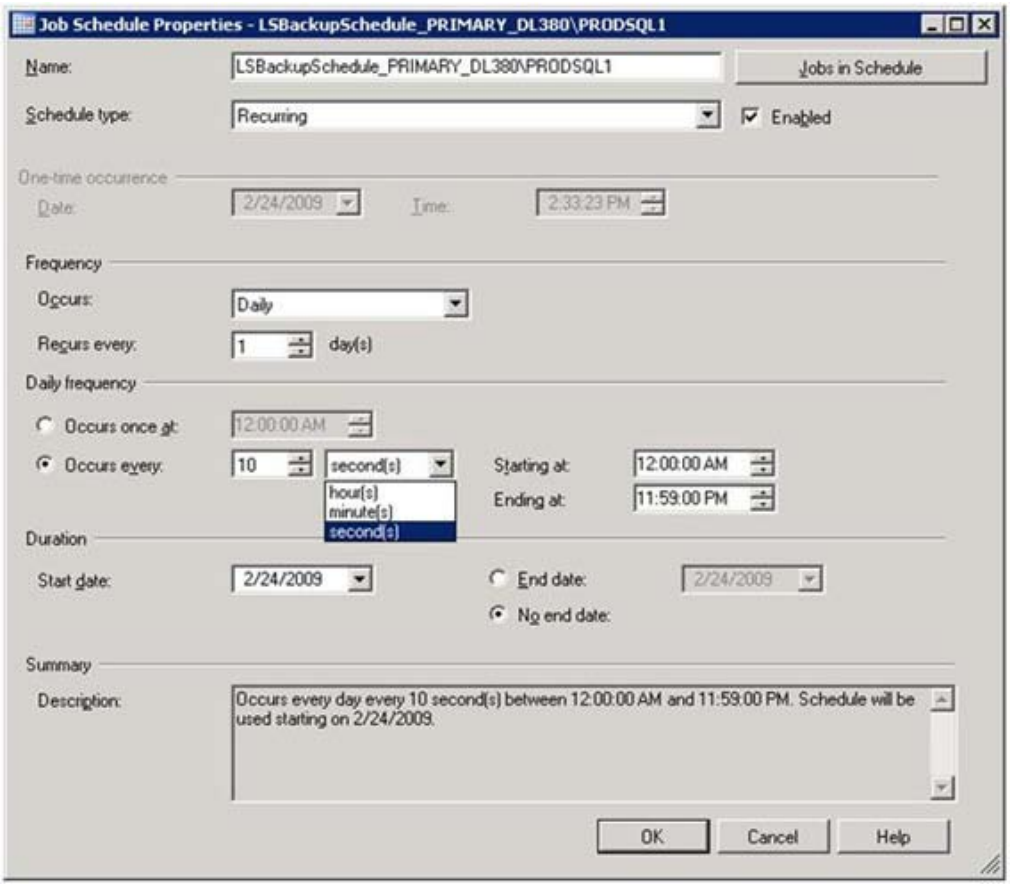

Figure 3: SQL 2008 Log Shipping Job Schedule Properties

Backup Compression in SQL Server 2008. SQL 2008 also supports backup compression, providing additional space and time savings via smaller backups and improved performance for most log shipping operations. Compressed backups become especially important when you are talking about frequent backups, sends, and restores of transaction log over the network; bandwidth can become an issue. The benefits of backup compression come with a cost, and that manifests itself in higher CPU utilization. Frequent backups and restores using compression will cause spikes in CPU utilization on both the primary and secondary servers. If the work load on the primary and secondary servers is not great, then frequent log shipping jobs can result in a much lower level of data loss in case of primary server failure.

Failover Clustering

Up to this point, we have talked about HA solutions that provide redundancy at the database level (database mirroring and log shipping). With Failover Clustering, SQL Server 2008 provides server-level redundancy for an entire instance of SQL Server leveraging Windows clustering found in the Enterprise Edition of the Windows Server operating system. In a failover cluster, multiple servers, also known as nodes, connect to shared disk resources. When SQL 2008 is installed on a failover cluster it appears as a single logical server on the network regardless of how many physical nodes or servers reside within the cluster. One node manages one particular SQL Server instance and associated services. If one node fails, another node in the cluster will take over its

responsibilities. The failover process is automatic, prevents data loss, and provides automatic client redirection. Failover from primary to secondary is not quite as fast as database mirroring. Windows Server 2008 has reduced the hardware requirements necessary to support clustering and is easier to set up and manage versus previous versions of the Windows operating system.

There are some things to keep in mind when considering a cluster solution. SQL 2008 has certain components that are "cluster aware" (database engine, full-text search, replication, and analysis services). These components can be clustered. SQL Server integration services and reporting services cannot be clustered. However, you can increase the availability of a reporting server using a scale out deployment where two or more report server instances share a single report server database (see SQL 2008 books on-line, **http://technet.microsoft.com/en-us/library/bb630407.aspx**).

Up to 16 nodes can be supported in a failover cluster running SQL 2008 Enterprise Edition on top of Windows 2008 Enterprise Edition. Two-node failover clustering is supported with the Standard Edition of SQL Server 2008.

Geographically Dispersed Failover Clustering

Windows 2008 and SQL 2008 support geographically dispersed failover clusters to provide server-level redundancy across sites. If the site, node, or disks fail, the redundant systems and disks enable the failover cluster to take over and handle subsequent activities on a separate site. Prior to Windows 2008 failover clustering, there were some limitations to geographically dispersed nodes for disaster recovery. The nodes needed to be part of the same LAN and subnet. There needed to be a heartbeat between nodes with less then 500ms latency or else the servers would assume failover, and nodes on the cluster had to share the same disk array. Microsoft has enhanced clustering with Windows 2008 and removed these limitations.

Peer-to-Peer Replication

There are many different ways to replicate data in SQL 2008. Replication allows you to copy complete databases or portions of a database from one SQL Server instance to another. Replication uses a publisher-subscriber model where a primary server (publisher), distributes data (articles) to one or more secondary servers (subscribers).

Peer-to-peer replication in SQL Server 2008 allows databases to replicate to each other. When using peer-to-peer replication, all partners in the replication topology are peers, and changes allowed on each database can be applied to other nodes in the group. Data can be updated on all databases configured in the peer-to-peer replication topology. If one of the peers is not available, then traffic can be redirected to another peer in the group. Applications must be designed to connect to another node in the group if the principal server is not available. Peer-to-peer replication is supported on both the Enterprise and Standard editions of SQL Server 2008.

SQL Server 2008 enhancements. With SQL 2008, peer-to-peer replication capabilities have been enhanced by enabling the addition of nodes to a replication topology while keeping replication online. In SQL Server 2005, replication had to be taken offline to add new nodes into the replication topology. Also, SQL Server 2008 introduced a new "Topology Viewer" to simplify the set-up, management, and monitoring of peer-to-peer replication. SQL Server 2008 also introduced conflict detection to help protect against accidental conflicts when multiple replication peers update the same row.

Database Snapshots

There are many different causes of downtime, but the most common cause is human error. Database snapshots, which were new in SQL Server 2005, enable quick discovery and recovery from user errors. Database snapshots are static, read-only views of a database at the moment of the snapshot, and multiple snapshots can exist on a source database at any one time. The snapshots will always reside on the same server instance as the database and exist until explicitly dropped by the database owner. In case of user error, you can roll back the source database to the state it was in when the snapshot was created.

If the database or server is offline or corrupted, then reverting will not work as the snapshot relies on the source database. Database snapshots are not a replacement for regular backups but do provide a quick way to revert back to changes made since the last snapshot. Database snapshots can also be used for reporting purposes without affecting the availability of the source database.

Dynamic Configuration

Dynamic configuration is one of the SQL Server 2008 components that enables access to hardware features, such as hot upgrades, while the system is still up and running. If your system supports hot-add memory, then memory can be added with the system still up, and SQL Server will automatically uses this memory through dynamic memory. SQL Server 2008 also supports another hardware feature called hot-add CPU, so that you can add processors with no interruption in activity. For any system that needs hot-adds to memory or processor, dynamic configuration supports this functionality and reduces downtime and increases availability.

Conclusion

SQL Server 2008 provides a variety of solutions to improve availability. In deciding which option or options make sense, it is important to consider a number of factors, including the effectiveness of the solution, which should be weighed against implementation, management, software/hardware costs. It is important to establish your organization's SQL Server availability goals and service-level uptime requirements (99.99% uptime?). Do you need to protect a single database or the entire SQL Server instance? Do you require automatic or manual failover? Do you require site-level protection? The answer to these questions will dictate the solutions.

SQL 2005 introduced a variety of high availability features, such as database mirroring and improved failover clustering. SQL Server 2008 has added on to the SQL 2005 architecture with some interesting new features, including page-level data recovery via Database mirroring, sub-minute log shipping, geographically dispersed failover clusters, and hot-adds for memory and processors. Microsoft has built on top of the High-availability features in SQL 2005 with an evolution of new features in SQL 2008 giving DBAs a richer tool set for designing a robust High-availability solution.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge. Check out the following Global Knowledge courses:

Implementing and Maintaining SQL Server 2008

Maintaining a Microsoft SQL Server 2008 Database

For more information or to register, visit **www.globalknowledge.com** or call **1-800-COURSES** to speak with a sales representative.

Our courses and enhanced, hands-on labs offer practical skills and tips that you can immediately put to use. Our expert instructors draw upon their experiences to help you understand key concepts and how to apply them to your specific work situation. Choose from our more than 700 courses, delivered through Classrooms, e-Learning, and On-site sessions, to meet your IT and management training needs.

About the Author

Michael Manning is a Global Knowledge instructor and consultant specializing in Microsoft SharePoint and SQL Server technologies. He has been working with SQL Server going back to the SQL 7.0 days and currently resides in Seattle, Washington.