ABSTRACT: In today’s highly competitive business environment, for most companies it is imperative that their business data is always accessible from their data servers in a seamlessly fault tolerant manner. DBMSs, which constitute the core of most business information systems, are attempting to implement this requirement through a feature often referred to in the data management literature as ‘high availability’. Against the backdrop of a leading DBMS, Microsoft SQL Server 2005, we define the concept of high availability and examine the various approaches to high availability, namely, mirroring, clustering, replication, and log shipping, which should be of interest to database administrators and the students of database management system technology.

Categories and Subject Descriptors
H.2.1[Database Management]; Modeling Techniques H.2.4 [Systems]; Relational Databases

General Terms
Database management system, SQL server, Business Information System

Keywords: Database administration, SQL, Data mirroring, Data Clustering, Data replication

Received 24 Dec. 2005; Reviewed and accepted 13 Mar. 2006

1. Introduction

One of the most important characteristics of a business information system is the availability of the data accessed by the system. Large corporate systems such as those used for retail point-of-sale processing require high availability. Their performance is measured against a standard known as the five 9’s – that is the system should be available 99.999% of the time. This standard is generally termed high availability and equates to about five minutes of downtime on a per year basis.

High availability is achieved through the use of redundant information systems and technical components that allow failover; that is, the switching from a failed system to a backup system that is ready to take over all processing immediately, preferably without any drop in system performance. High availability has long been a selling feature for the larger database management system products such as Oracle’s Oracle RDBMS and IBM’s DB2. For example, Oracle currently touts the grid computing features of its Version. Traditionally high availability has been achieved only by paying a significant price for the requisite hardware and software. Obviously, the value added by high availability systems has to exceed the cost of those systems, and so only organizations that enjoy a significant additional revenue stream are able to provide systems that meet the 99.999% high availability standard.

Microsoft’s SQL Server is an excellent RDBMS and it carries a relatively inexpensive price tag when compared to other RDBMS products such as Oracle and IBM’s DB2. In the past, information technology managers have questioned the ability of SQL Server to deliver high availability for very large systems – those databases where data storage requirements are measured in terabytes. The term very large database (VLDB) is often used to refer to these giant stores of data.

The recent release of Microsoft’s SQL Server 2005 will cause many organizations to reevaluate the potential for SQL Server in terms of its capacity to provide high availability for VLDB systems. Why? Quite simply, SQL Server 2005’s smaller price tag makes it a relatively inexpensive solution when compared to Oracle and DB2. Large organizations can benefit by lowering the cost of failover if they can achieve required performance standards such as system response time. Smaller organizations will be able to build and provide high availability systems for the first time.

Already some firms are moving in this direction. AIM Healthcare Services located in Franklin, Tennessee has a nine terabyte SQL Server 2005 database that supports its online transaction processing (OLTP) systems. Similarly, Barnes & Noble Inc. has a three terabyte data warehouse running on a 64-bit SQL Server 2005 platform. The NASDAQ Stock Market Inc organization has replaced its Tandem mainframe with a SQL Server 2005 database that disseminates market trade data processing 5,000 transactions per second and 100,000 queries per day while storing eight million new data rows daily (Lai 2005)

Database administrators will be interested in the SQL Server 2005 features that support high availability. We will examine the key features of mirroring and clustering. These features are compared to other approaches to high availability such as replication and log shipping that have been in use with earlier versions of SQL Server.

2. Mirroring

2.1 What is Mirroring?

Database mirroring involves the use of multiple servers with duplicate copies of data. Figure 1 displays a three-server scenario adapted from Heiges-2 2005. Here the SQL Server instance running on Server A performs the role of a Principal server while the SQL Server instance running on Server B performs role of a Mirror server. An SQL Server instance running on Server C fulfills an optional Witness server role to act as a witness to mirroring transactions in the event that failover is necessary. The SQL Server production database located on Server A has an up-to-date copy of the database stored on server B. If failover occurs, it will appear that recovery has taken place because Server B will take over responsibility for on-line transaction processing. This failover can be automated and should occur within about 3 seconds (Heiges-2 2005).

Database mirroring with SQL Server 2005 has certain advantages including:

- Failover can be automated and takes place quickly.
- The Principal and Mirror servers do not need to be geographically close to one another—they can be located anywhere in the world.
- SQL Server 2005 does not require the server hardware to be from a special hardware capability list.
- Disaster recovery is enabled by using the Mirror server as a warm standby server.

2.2 How Does Mirroring Work?

Mirroring is best understood in terms of the Principal, Mirror, and Witness roles. The server acting as the Principal is a typical production server that serves applications that process business transactions. In Figure 1, Server A (the Principal) handles transactions through Switch #1. As transactions are processed, the Principal transmits a copy of each logical transaction to the Mirror on Server B and waits for acknowledgement from the Mirror prior to committing each transaction. The Mirror receives transactions from the Principal and records them to its log file. It then applies the transactions to its
copy of the database through use of the redo process normally used for database transaction recovery. In essence, the Mirror is redoing transactions on a continuous basis. Mirroring differs from database replication because database mirroring takes place at the level of the physical log record while replication works at the logical level.

The Witness (Server C in Figure 1) monitors the mirroring process. It has the sole task of participating to form a quorum in the event that automatic failover needs to be initiated. The Witness server neither processes transactions nor stores any data for the production or mirrored database. While Figure 1 shows the Witness (Server C) as a separate server, in fact, this SQL Server instance can be on the same physical server as either the Principal or Mirror processes (Heiges-2 2005), although use of a third server reduces the failure points in the system.

All transaction processing from client computers are handled by the Principal server. Access to the Mirror server by client applications does not take place unless a failover occurs. When a failover occurs, the Mirror server assumes the Principal role. At the time of failover, the Witness participates to form a quorum as two servers are required by the SQL Server software in order to determine (by voting) which server in the system will perform the Principal role. This eliminates the possibility of two servers both attempting to fulfill the Principal role. If the Witness server fails, automatic failover cannot take place; however, a database administrator can still initiate a manual failover to the mirrored database (Heiges 2005).

### 2.3 Operating Modes
Database mirroring can be accomplished through one of three modes of operation. These modes are termed: (1) high availability, (2) high protection, and (3) high performance. A database administrator can move a database from one mode to another by determining whether or not to use an SQL Server instance in a Witness role, and by using SQL Server Management Studio software to set a safety switch setting for the mirroring session to either full (for synchronous communication) or off (for asynchronous communication).

When in asynchronous communication, transactions commit on the Principal server. When a failover occurs, the Mirror server assumes the Principal role. If the Witness participates to form a quorum as two servers are required by the SQL Server software in order to determine (by voting) which server in the system will perform the Principal role. This eliminates the possibility of two servers both attempting to fulfill the Principal role. If the Witness server fails, automatic failover cannot take place; however, a database administrator can still initiate a manual failover to the mirrored database (Heiges 2005).

#### 2.3.1 High Availability Mode
The high availability operating mode requires a Witness in order to support automatic failover. Data communication is through synchronous operation. The safety switch setting is full. If the Principal (Server A in Figure 1) experiences failure, then Server B, in consultation with the Witness, will become the Principal by having the system failover to Server B. When Server A is restored, it becomes the Mirror and processing can continue uninterrupted.

Recall from the earlier discussion that one of the advantages of mirroring is that the hardware components for the servers can be different. In fact, Server A may be a newer, more powerful server than Server B. Thus, a database administrator may, once Server A is restored, wish to intervene by initiating a manual failover so that Server A again becomes the Principal. This will restore the processing efficiency level to pre-failover limits since Server A is the more powerful machine.

#### 2.3.2 High Protection Mode
In the high protection operating mode, the safety switch is off and there is no instance of SQL Server performing the Witness role. Again, data communication is through synchronous operation. The Principal still waits for the Mirror to report the receipt and logging of transactions before committing them to the database. The only difference from the high availability mode is that automatic failover is not supported. A database administrator must initiate a manual failover.

#### 2.3.3 High Performance Mode
In the high performance operating mode, the safety switch is off and there is no instance of SQL Server performing the Witness role. Data communication is asynchronous. Transactions are processed by the Principal without waiting for confirmation that the Mirror has received and logged copies of the transactions. Some data loss can occur because the Mirror server may not be able to keep up with all of the processing, and transactions can commit on the Principal that have not committed on the Mirror. The potential data loss is greatest when the Principal has a heavy work load or if the Mirror is overloaded, perhaps with additional applications executing on that machine.

### 2.4 Other Issues
It appears that the best mode to use is a high availability mode; however, this mode is designed for use with a network that has either a dedicated connection or a simple network configuration that reduces the possibility of network failures (Setting 2005). This type of network is also recommended for both high protection and high performance modes, but these two modes are not affected by network reliability to the extent that high availability mode is affected (Setting 2005).

Additionally, if three separate servers are not available for high availability mode, the Witness role can be performed by an additional SQL Server instance on either the Principal or Mirror server. Also, the Witness instance of SQL Server can be run through the use of the currently free version of SQL Server Express (Heiges 2005).

### 2.5 Deployment Guidelines
Some of the specific deployment guidelines provided by Microsoft are as follows (Setting 2005).

- Run in asynchronous, high-performance mode initially as this mode is not as sensitive to the network environment and provides a configuration for testing mirroring for bandwidth capacity.
- If synchronous mode works well, synchronous operations can be explored in order to improve data protection; however, running in synchronous operation mode can result in slow performance if the database applications generate large amounts of transaction log data.
- Switch to high availability mode in order to use automatic failover once it has been determined that the high protection mode does not lead to excessive network error failures.

### 3. Clustering

#### 3.1 What Is Clustering?
In the past, clustering has always been the first choice for high availability solutions. While the mirroring of databases occurs at the database level, clustering focuses on maintaining high availability through the management of a set of servers at the systems level. A clustered set of servers share a set of disk drives. The failure of a server results in a reallocation of its defined resources, such as disk drives and SQL Server instances, to another server. How long does it take for this failover to occur? The time varies based on hardware and takes at least 30 seconds though it is often the case that a failover can take several minutes (Heiges-2 2005).

Clustering is often perceived to be a means of improving performance in a Windows environment by enabling the scaling of demands on processor resources. This is simply not the case—clustering simply enables failover of defined services and shared data stores.
Performance improvements through scaling require the use of distributed partitioned views or replication, or both (Knight 2005). Clustering has an additional limitation. While database mirroring allows the Principal and Mirror to be located geographically anywhere in the world, clustering requires the clustered nodes to be within about 100 miles of each other as a maximum on distance separation (Heiges-2 2005). System performance is best when the nodes are located closer to one another.

3.2 Software Requirements
To build a cluster of servers in a Windows environment requires at least Windows 2003 Enterprise Edition. Clustering can also be accomplished through the Windows 2003 Data Center, but at a significant increase in price. The Microsoft Cluster Service (MSCS) is the main software tool that controls Windows clustering. Failover clustering is supported in SQL Server 2005 Enterprise Edition, Developer Edition and, with some restrictions, Standard Edition (Configuring 2005).

Each server in a cluster is termed a node. Both Windows 2003 Enterprise Edition and Windows 2003 Data Center Edition support eight-node clusters, an improvement over the two-node limitations of the Windows 2000 Enterprise Edition. The system administrator can use the Cluster Administrator software to manage clustered resources such as SQL Server Agent. This tool enables specifying the preferred owner of a resource as well as specifying what are termed possible owners for resources. This enables specifying the node to which services will failover. Applications connected to SQL Server by use of a virtual server name so that the server name can be reallocated as necessary in the event of failover.

3.3 Hardware Requirements
Clustering requires that the hardware be on Microsoft’s hardware compatibility list. In order to share disk resources, the system requires a shared drive array such as a shared SCSI device or a stored area network (SAN) device. It is recommended that cards required for SAN connectivity be purchased in pairs so as to avoid creating a server as a single point of failure (Knight 2005).

3.4 Cluster Types
Windows clustering can be accomplished by use of either a single-node cluster or multi-node cluster. The single-node cluster has been termed active/passive clustering because the cluster had two nodes, one that is active and one that is passive. The failure of an active node results in the passive node becoming active and assuming control of all specified cluster resources. The passive node is not accessible unless it becomes the active node.

Multi-node clusters have been termed active/active clusters because all nodes are accessible. The failure of a node causes its resources to be shifted to another active node. This will influence the selection of hardware so as to avoid any decrease in system response that could occur if a less powerful active node acquires resources and application support responsibilities formerly assigned to a more powerful node (Knight 2005).

Multi-node clusters have an SQL Server instance installed/running on each node. Consider a situation with two nodes in a cluster. Opening the Windows control panel’s services option for Node 1 will display two SQL Server services – one each for Node 1 and Node 2 with only the Node 1 SQL Service service turned on. Likewise, the Windows control panel’s services option for Node 2 will display two SQL Server services, but the one for Node 1 will be turned off while the one for Node 2 will be turned on. If Node 1 fails, then the SQL Server instance for Node 1 will be turned on for the computer functioning as Node 2.

4. Replication
4.1 What Is Replication?
Replication is a strategy for distributing data or data subsets across multiple servers through use of a publish-subscribe model. The primary server is designated the publisher and it distributes data to one or more secondary servers designated as the subscriber(s). Replication is flexible in that it supports real-time availability and scalability across the servers. Data that is replicated to specific subscribers can be filtered and partitioned updates are supported so that the subscriber database does not have to be a complete copy of the database being replicated. All editions of SQL Server 2005 except for SQL Server Express and SQL Server Mobile support replication.

Replication enables scalability and availability of data through the maintenance of copies of data that are continuously updated across multiple servers. The database administrator determines how many servers are required to support scalability adequately. When a server is down, data queries can be routed to a different server. Application requirements will affect the approach taken by a database administrator (Improving 2005).

Applications supporting primarily querying or read activities are best supported by data caching where data is replicated to multiple servers. When a single database stores data, problems with data availability can be encountered as the load on a database increases. Client data requests can be scaled across multiple read-only databases to scale out the read portion of the work load as illustrated in Figure 2.

Applications that must be robust to planned and unplanned database downtime can be supported through the replication of data to a warm standby server. The standby server will support data requests during downtime for the primary server. At times the data replicated may be a subset of the data stored on the primary server. However, if the entire database needs to be available, then database mirroring is more efficient than replication and should be the high availability approach selected by a database administrator.

One shortcoming of replication is that the replication of data results in a higher latency of data distribution than mirroring, and the replicated database is not an exact copy of the original database. Replication supports protection at the table level so data subsets are easily protected, and this is a significant advantage of replication. SQL Server 2005 supports three types of replication: (1) snapshot, (2) transactional, and (3) merge. Of these, transactional replication offers the lowest latency of data distribution.

4.2 Standard Transactional Replication
The standard approach to transactional replication assumes subscribers are read-only. Usually, a single publisher transmits update data to the subscribers. The replication topology for the relationship between a publisher and subscriber is hierarchical and can include the use of republishing subscribers that receive updates and further transmit these updates to what are termed leaf-node subscribers.

4.3 Peer-to-Peer Replication
The new peer-to-peer replication supported in SQL Server 2005 supports applications that both read and modify data in any of the databases that are part of the replication. A typical application might be an on-line order-entry system. Applications are programmed to route data requests and updates to alternative servers should one server become unavailable as each server contains identical copies of data. The use of multiple nodes improves read performance. All of the nodes are termed peers as each node publishers/subscribes the same schema and data (Peer-to-Peer 2005).
5. Log Shipping

5.1 What is Log Shipping?

Log shipping involves shipping transaction log backups from a primary database to one or more secondary databases. These databases should reside on separate servers. The transaction log backups are restored to a secondary database(s) and this enables a database administrator to keep the secondary database(s) closely synchronized with the primary database. When multiple primary databases are in use, an organization can achieve improved resource utilization by having a secondary server perform the backup role for multiple primary databases. Additionally, the log shipping process can be automated by enabling Log Shipping on the respective SQL Server instances.

Sometimes an optional third server is used to monitor the history and status of these backup and restore operations. This monitor server should be separate from any primary/secondary servers as this minimizes the possibility of losing information. Additionally, a monitor server can perform these functions for numerous primary and secondary servers. This monitor server can publish alerts to a database administrator if the backup/restore operations fail. The log shipping operations are summarized in Table 1 and are described in more detail in the SQL Server 2005 MS SQL Help (Understanding 2005).

One shortcoming of log shipping is that failover does not automatically occur from the primary to secondary server. Secondary databases must be brought online manually by a database administrator. A second shortcoming is the time required for the process of restoring transaction log backups. Restoration can consume as much as 10 minutes or more and this does not satisfy the five 9’s standard for high availability. Another shortcoming is that some data transactions processed on the primary database may be lost in the event of failure because the batch nature of the backup/restore operation means that the secondary database(s) are not completely synchronized with the primary database. Clearly, mirroring is a better approach for critical data. Still, some organizations use this approach because it is available in SQL Server 2000 Enterprise Edition and can be implemented through the use of scripts for the Standard Edition.

5.2 Initializing Log Shipping

Secondary databases are initialized by first taking a full backup of a primary database, then restoring this backup to the secondary database through use of either the NORECOVERY or STANDBY option. A database administrator can accomplish this manually or through use of the SQL Server 2005 Management Studio software. How often should the transaction log backups be applied to the secondary server? This depends on how frequently transaction log backups are created for the primary server. The default run for the Log Shipping Copy job by an SQL Server Agent is every two minutes. However, it should be obvious that more frequent backup/restore operations to the secondary server will tend to both reduce the work required to bring a secondary server online and minimize data loss. Minimizing downtime is the critical objective for high availability.

The detailed process for failing over to a log shipping secondary server is described in a Microsoft MSDN web page (Failing 2005).

6. Summary

Table 2 summarizes the critical features of the four approaches to high availability (Heiges-2 2005). Both mirroring and clustering support automatic failover with clustering supporting shared storage. Clearly mirroring is best in supporting the high availability standard with the minimum failover time and no data loss along with no geographical limitation. In summary, the addition of the mirroring capability to Microsoft SQL Server 2005 and the increase of cluster sizes to eight servers from the limitation of two clusters with earlier versions of SQL Server significantly improve the potential for SQL Server 2005 to meet the needs of ever larger organizations. It also brings the potential for supporting high availability systems to smaller organizations. The topic should be covered and examined in detail in courses of study that focus on the delivery of systems based on database management system technology.

References


Douglas B. Bock is Professor of Computer Management and Information Systems at Southern Illinois University Edwardsville where his teaching specialty focuses on database administration and database programming. His doctorate is from Indiana University and he is also a Microsoft Certified Solution Developer for .NET technology.