An Introduction to SQL Server Clustering

If your mission-critical SQL Server should experience a motherboard failure, how long will it be down? One hour, four hours, a day, or longer? How much will this cost your business in lost sales or productivity? And perhaps even more important to you, what will it do to your stress level?

Being a SQL Server DBA can be demanding and stressful, especially as the success of your company is often a function of your SQL Server's uptime. While we, as DBAs, have some control over the uptime of our SQL Servers, we don't have full control. There is not much we can do if a motherboard fails on a server, other than be prepared.

As you may already be aware, there is one way to help boost your SQL Server's uptime, and that is by clustering SQL Servers. This way, should one SQL Server fail in the cluster, another clustered server will automatically take over, keeping downtime to minutes, instead of hours or more.

The purpose of this article is to introduce you to SQL Server clustering, along with its pros and cons. If you are considering clustering SQL Server to help reduce potential downtime, this article is a good place to start.

What is Clustering?

Clustering can be best described as a technology that automatically allows one physical server to take over the tasks and responsibilities of another physical server that has failed. The obvious goal behind this, given that all computer hardware and software will eventually fail, is to ensure that users running mission-critical applications will have little or no downtime when such a failure occurs. Downtime can be very expensive, and our goal as DBA is to help reduce it as much as possible.

More specifically, clustering refers to a group of two or more servers (generally called nodes) that work together and represent themselves as a single virtual server to a network. In other words, when a client connects to clustered SQL Servers, it thinks there is only a single SQL Server, not more than one. When one of the nodes fails, its responsibilities are taken over by another server in the cluster, and the end-user notices little, if any differences before, during, and after the failover.

Microsoft added clustering features to its operating system when they introduced Windows NT Server 4.0 Enterprise Edition several years ago. The actual clustering feature was called MSCS (Microsoft Clustering Server). While some brave folks actually put the software into production, I personally avoided it as it was not as dependable as Microsoft led you to believe. Also, about this same time, SQL Server 6.5 Enterprise Edition was released, allowing it to be clustered. This was a very crude attempt at clustering SQL Server that was rarely implemented in the real world.

Later, when SQL Server 7.0 became available, there were major improvements in SQL Server clustering, but no way was it still good enough, as Windows NT 4.0 Server Enterprise MSCS was still being used at its foundation, and it just wasn't good enough for maintaining high-availability servers.

Fortunately, Microsoft's second attempt at clustering, now called the Microsoft Cluster Service, in Windows 2000 Advanced Server and Windows 2000 Datacenter Server, is much better. While I won't claim it is perfect, at least I now am willing to use it to cluster mission-critical SQL Servers. Cluster Service not only works well when clustering SQL Server 7.0, it is even better if you want to cluster SQL Server 2000, which has greatly enhanced clustering support.

One very important aspect of clustering that often gets overlooked is that it is not a complete backup system for your applications. It is only one part of a multi-part strategy required to ensure minimum downtime and 100% recoverability.

The main benefits that clustering provides is the ability to recover from failed server hardware (excluding the shared disk) and failed software, such as failed services or a server lockup. It is not designed to protect data, to protect against a shared disk array from failing, to prevent hack attacks, to protect against network failure, or to prevent SQL Server from other potential disasters, such as power outages or acts of God.

Clustering is just one part of an entire strategy needed to help reduce application downtime. You will also need to purchase a shared disk array (more on this later) that offers redundancy, make tape backups, put the server behind a firewall, make sure your network connections have redundancy, use battery backup, and locate the server in a secure facility, among many other steps you can take. So don't think that clustering is all you need for creating a highly available SQL Server. It is just one part.

What Are the Types of Clustering?

When you decide you want to cluster SQL Server, you have a choice of configuring what is called Active/Active or an Active/Passive cluster. Each has its own pros and cons. Let's look at each, in the context of a two-node SQL Server cluster.

An Active/Active SQL Server cluster means that SQL Server is running on both nodes of a two-way cluster. Each copy of SQL Server acts independently, and users see two different SQL Servers. If one of the SQL Servers in the cluster should fail, then the failed instance of SQL Server will failover to the remaining server. This means that then both instances of SQL Server will be running on one physical server, instead of two.

As you can imagine, if two instances have to run on one physical server, performance can be affected, especially if the server's have not been sized appropriately.

An Active/Passive SQL Server cluster refers to a SQL Server cluster where only one instance of SQL Server is running on one of the physical servers in the cluster, and the other physical server does nothing, other that waiting to takeover should the primary node should fail.

From a performance perspective, this is the better solution. On the other hand, this option makes less productive use of your physical hardware, which means this solution is more expensive.

Personally, I prefer an Active/Passive configuration as it is easier to set up and administer, and overall it will provide better performance. Assuming you have the budget, this is what I recommend.

Two- or Four-Node Clustering?

SQL Server can be clustered using two nodes (using Windows 2000 Advanced Server), or it can be clustered using more than two nodes (using Windows 2000 Datacenter). Since I don't personally have any experience is three or four node clustering, I won't be discussing it here. But for the most part, what I say about two-node clustering also applies to three- or four-node clustering.

How Does Clustering Work?

Clustering is a very complex technology, so I will focus here on the big picture. In a two-cluster node, one of the SQL Servers is referred to as the primary node, and the second one is referred to as the secondary node. In an Active/Passive cluster design, SQL Server will run on the primary node, and should the primary node fail, then the secondary node will take over.

When you build a two-node cluster using Windows 2000 Advanced Server and Microsoft Clustering Service, each node must be connected to a shared disk array using either SCSI cables or fibre channel.

Typically, this shared disk array is a stand-alone unit that houses a RAID 5 or RAID 10 disk array. All of the shared data in the cluster must be stored on this disk array, otherwise when a failover occurs, the secondary node in the cluster cannot access it. As I have already mentioned earlier, clustering does not help protect data or the shared disk array that it is stored on. Because of this, it is very important that you select a shared disk array that is very reliable and includes fault tolerance.

Besides both servers being connected to a shared disk array, both nodes of the cluster are also connected to each other via a private network. This private network is used for each node to keep track of the status of the other node. For example, if the primary node experiences a hardware failure, the secondary node will detect this and will automatically initiate a failover.

So how do clients who are accessing SQL Server know what to do when a failover occurs in a cluster? This is the cleverest part of Microsoft Cluster Service. Essentially what happens in a SQL Server cluster is that you assign SQL Server its own virtual name and virtual TCP/IP address. This name and address is shared by both of the servers in the cluster.

Typically, a client will connect to the SQL Server cluster using the virtual name used by the cluster. And as far as a client is concerned, there is only one physical SQL Server, not two. Assuming that the primary node of the SQL Server cluster is the node running SQL Server on an Active/Passive cluster design, then the primary node will respond to the client's requests. But if the primary node fails, and failover to the secondary node occurs, the cluster will still retain the same SQL Server virtual name and TCP/IP address, although now a new physical server will be responding to client's requests.

During the failover period, which can last several minutes (the exact amount of time depends on the number and sizes of the databases on SQL Server, and how active they are), clients will be unable to access SQL Server, so there is a small amount of downtime when failover occurs.

How the client software reacts to the failover process depends on the software. Some software will just wait the failover out, and when the failover has completed, it will continue just as nothing had happened. Some software will present a message box on the screen, describing a lost connection. Other client software will not know what to do, and users may have to exit, and then reload the client before they can access SQL Server again.

As part of the testing process when implementing a SQL Server cluster, it is important to find out how all of the client software that connects to SQL Server reacts to a failover. This way, you can inform your users of what to expect, so they are better able to deal with it when it does occur.

Once a failover occurs, you will want to find out what caused the failover, and then take the necessary action and correct the problem. Once the problem has been fixed, the next step is to failover SQL Server back to the primary node from the secondary node. You can schedule to do this anytime, preferably when user activity is light on the system.

What are the Pros and Cons of Clustering?

Implementing SQL Server clustering is a big decision, and one fraught with many gochas. Before you undertake such a large and important project, you will want to carefully evaluate the pros and cons of clustering, which include, but are not limited to:

Pros of SQL Server Clustering

- Reduces downtime to a bare minimum.
- Permits an automatic response to a failed server or software. No human intervention is required.
- It allows you to perform upgrades without forcing users off the system for extended periods of time.

- It allows you to reduce downtime due to routine server, network, or database maintenance.
- Clustering doesn't require any servers to be renamed. So when failover occurs, it is relatively transparent to end-users.
- Failing back is quick, and can be done whenever the primary server if fixed and put back on-line.
- In some cases, clustering can be used to increase the scalability of an application. For example, if a current cluster is getting too busy, another server could be added to the cluster to expand the resources and help boost the performance of the application.

Cons of Clustering

- More expensive than other failover alternatives, such as log shipping or stand-by servers.
- Requires more set up time than other alternatives.
- Requires more on-going maintenance than other alternatives.
- Requires more experienced DBAs and network administrators.

Software Needed for Clustering

The software you need for SQL Server clustering depends on whether you want to cluster two nodes, or more than two nodes.

To cluster two nodes, you will need the following:

- Two Microsoft Windows 2000 Advanced Server Licenses
- One SQL Server 7.0 Enterprise or SQL Server 2000 Enterprise Licenses for Active/Passive, or two licenses for Active/Active
- The latest Windows 2000 and SQL Server Service Packs

To cluster more than two nodes, you will need the following:

- Two or More Microsoft Windows 2000 Datacenter Server Licenses
- Two or More SQL Server 7.0 Enterprise or SQL Server 2000 Enterprise Licenses

- The latest Windows 2000 and SQL Server Service Packs

I want to emphasize that you always want to go with the latest service packs, as many irritating cluster-related bugs have been fixed by them.

Hardware Needed for Clustering

Assuming you are clustering two SQL Servers, you will need at the very minimum the following:

- Two servers with a minimum of 256MB RAM and a single Pentium III CPU.
- One shared disk array that supports RAID 5 or 10, either SCSI or fibre channel.
- Each server must have at least one local SCSI hard disk on its own SCSI controller.
- Each server must have a SCSI or fiber channel adapter to talk to the shared disk array. The shared disk array cannot use the SCSI controller used by the local hard disk or CD-ROM.
- Each server must have two PCI network cards (one for the private connection and one for the public connection).

How you size the physical servers (CPUs, RAM, amount of shared disk array) is very similar to how you would size a non-clustered server if you plan to use an Active/Passive configuration. But, if you intend to use an Active/Active configuration, then ideally each physical server needs to be sized to run all instances of SQL Server that run on both the primary and secondary nodes, should failover occur and both instances of SQL Server have to run on the same physical server.

Ideally, both physical servers should be identical in hardware, drivers, software, and configuration. There are some exceptions to this allowed, but I would not recommend making them. The closer each physical server is to each other, the less problems you will have.

Another very important consideration when selecting clustering hardware is that is must be on Microsoft's Hardware Compatibility List (HCL) as a supported system. What do I mean by a supported system? Before Microsoft will support your cluster, all of the cluster hardware you select (servers, cards, shared array, etc.) must have been tested as an entire system and approved by Microsoft. If the system you purchase isn't an approved system, then Microsoft will not help you if you call them, even if you pay them for support.

Even if all of the individual components of your cluster system have been approved individually by Microsoft for clustering, if they have not all been tested as a single system, then you will still not receive any support. That's why you need to check out the HCL carefully before you order your equipment to ensure that you purchase an approved clustering system.

Unfortunately, Microsoft's HCL website is often outdated. What this means is that as new equipment comes out (and we always seem to want to use the latest and the greatest), it may not be added to their approved cluster systems for months. You can of course take the risk of selecting equipment that is not part of an approved Microsoft system (hoping that it will eventually be tested and certified by Microsoft), but should you have a problem and call Microsoft, you risk not getting any help from them.

Setting Up and Managing a SQL Server Cluster

This topic could fill a book, but unfortunately, there is not enough room here for that. So what I will do here is just cover the basics so you have some feel for what you face.

Based on my experience with SQL Server clustering, you should keep the following in mind:

- Buy an approved cluster system and have it delivered to you 4-8 weeks before you plan to implement it. This will give you lots of time to learn about it and work out any problems. I guarantee you, you will have problems.
- Consider attending training on clustering, or at least bringing in a consultant in early in the game to help you plan your implementation. If you have never done this before, you will need all the help you can get. Unfortunately, there is not a lot of good information available on clustering, so you will have to fend for yourself more than you usually do.
- Once your cluster is up and running, test it thoroughly as possible using the same databases and clients you intend to use on it when you start production.
- Thoroughly document everything as you build your cluster.
- Develop a plan to regularly monitor the cluster.
- Develop a plan to regularly test the cluster to ensure that it will failover as expected.
- Develop a formal backup and recovery plan, and test it.

- Create a disaster recovery plan for what you intend to do should you loose all the nodes in your cluster.

Implementing and managing a cluster is complex. You will want to assign a team of top network and database administrators to implement and manage it.

Is Clustering for You?

Hopefully, this article has provided you with some additional information on SQL Server clustering that you didn't have before. And I hope I haven't scared you away from SQL Server clustering.

While SQL Server clustering is not an easy challenge, it is often a worthwhile one. After installing two SQL Server clusters myself, and currently working on a third, I feel that SQL Server clustering is very valuable for many organizations, and in fact it has reduced my stress own level somewhat. Now I don't have to worry (as much) about extended periods of down time. Now that the clusters are up, they haven't presented any problems and are purring away.

Madhusudhana Rao Gadipudi