Choosing a High-Availability Solution

High availability has become an increasingly popular subject in SQL Server. Not only have there been entire books dedicated to high availability, but we have also seen specialized books written on each topic falling under the high-availability umbrella. We will not be going into great detail here, but it is important to understand how each of SQL Server's high-availability solutions can play a role in your environment. This chapter gives an overview of each solution to differentiate what factors play a role in designing the best, overall solution for your organization. By the end of this chapter, you should be armed with the information necessary to make an informed decision on the best usage scenarios for each high-availability technique offered in SQL Server 2012.

What Exactly Is High Availability Anyway?

First, it is important to understand what high availability actually means. The terms high availability and disaster recovery (DR) are often confused or thought of as the same thing. Just because you have implemented a high-availability solution does not mean you are prepared for a disaster. High availability covers hardware or system-related failures, whereas disaster recovery can be used in the event of a catastrophic failure due to environmental factors. Although some of the high-availability options may help you when designing your DR strategy, they are not the be-all and end-all solution.

The goal of high availability is to provide an uninterrupted user experience with zero data loss, but high availability has many different meanings, depending on who you ask. According to Microsoft’s SQL Server Books Online, “a high-availability solution masks the effects of a hardware or software failure and maintains the availability of applications so that the perceived downtime for users is minimized.” (For more information, see http://msdn.microsoft.com/en-us/library/bb522583.aspx.) Many times users will say they need 100% availability, but what exactly does that mean? Does being 100% available mean data is 100% available during business hours, Monday through Friday, or data is available 24 hours a day 7days a week? High availability is about setting expectations and then living up to them. That’s why one of the most important things to do when dealing with high availability is to define those expectations in a Service Level Agreement (SLA) agreed on and signed by all parties involved.

Some of the things the SLA should cover are maintenance windows, the amount of recovery time allowed to bring the system back online because of catastrophic failure, and the amount of acceptable data loss, if any. Defining a maintenance window allows you to apply service packs, patches, and upgrades to the system to ensure optimal performance and maintain a supported version of SQL Server, as well as the operating system. Having a maintenance window allows you to do this in a tested and planned fashion. A drop-dead time should be determined, so that a back out plan can be executed if problems are encountered, ensuring system availability by the end of the maintenance window.

Defining the amount of time allowed to recover from a disaster, along with the maximum allowed data loss, will help determine what techniques you may need to use to ensure your SLAs are met. Every organization wants 100% availability 100% of the time; but when presented with the cost of a system that
could even come close to achieving this goal, organizations are usually willing to negotiate attainable terms. It is important to have an understanding of what it means for a system to be unavailable. Is it a minor inconvenience, because users within the organization will not be able to log their time, or is the organization losing thousands of dollars in revenue every hour the system is down? Answering these kinds of questions allows you to justify the cost of an appropriate solution. Each high-availability method brings unique characteristics to the table, and, unfortunately, there is no cookie-cutter solution. In the following sections, we discuss the individual techniques used to achieve your high-availability needs.

**Failover Clustering**

Failover clustering is a technique that uses a group of SQL Server instances residing on different servers to protect against failure of the instance currently serving your users. Failover clustering is based on a hardware solution comprised of multiple servers (nodes) that share the same disk resources. One server is active and owns the database. If that server fails, then another server in the cluster takes ownership of the database and continues to serve users.

**Key Terms**

When discussing high availability, each technique has its own set of key terms. At the beginning of each section, we list the terms used for each solution. Here are some of the terms you need to be familiar with when setting up a failover cluster:

- **Node.** A node is a server that participates in the failover cluster.
- **Resource group.** A resource group is a shared set of disks or other like resources grouped together to act as a single working unit.
- **Active node.** An active node is the node that has ownership of a resource group.
- **Passive node.** A passive node is a node waiting to take ownership of a resource group should the active node fail.
- **Heartbeat.** Heartbeats are health checks sent between nodes to ensure the availability of each node.
- **Public network.** The public network is the network used to access the failover cluster from a client computer.
- **Private network.** The private network is the network used to send heartbeat messages between nodes.
- **Quorum.** Quorum is a special resource group that holds information about the nodes, including the name and state of each node.

**Failover Clustering Overview**

You can use failover clustering to protect an entire instance of SQL Server. Although the nodes share the same disks or resources, only one server may have ownership (read and write privileges) of the resource group at any given time. If a failover occurs, ownership is transferred to another node and SQL Server is back up in the time it takes to bring the databases back online. The failover usually takes anywhere from a few seconds to a few minutes, depending on the size of the database and the types of transactions that may have been open during the failure.