Microsoft | Hosting

Business Continuity for Hosted SQL Server Using AlwaysOn Solutions

Introduction

Hosted Service Providers are under increasing pressure to provide access to customer data. Customers need data access globally, on demand, 24x7, with allowable unscheduled downtime in the single-digit hours per year, despite any kind of system outage, from a single server to a natural disaster. Microsoft® SQL Server® 2012 provides the hoster with new possibilities for achieving the required 9s for their customers.

This paper discusses how the hosting company can provide the customer with the protection and performance they desire and considerations for hosting company administrators in implementing AlwaysOn high availability and disaster recovery solutions.

High availability (HA) may be defined in a service-level agreement (SLA) as the percentage of uptime a hosting company guarantees a customer; minimizing unscheduled downtime, with an agreed-upon sacrifice in performance, if necessary, to keep multiple servers in sync.

Disaster recovery (DR) may be defined as a plan the hosting company uses with the customer's consent to recover from a data center failure, when access to a data center is cut off because of a power failure, natural disaster, etc. The customer's data must be brought back online from a different data center as quickly as possible with as little data loss as possible.

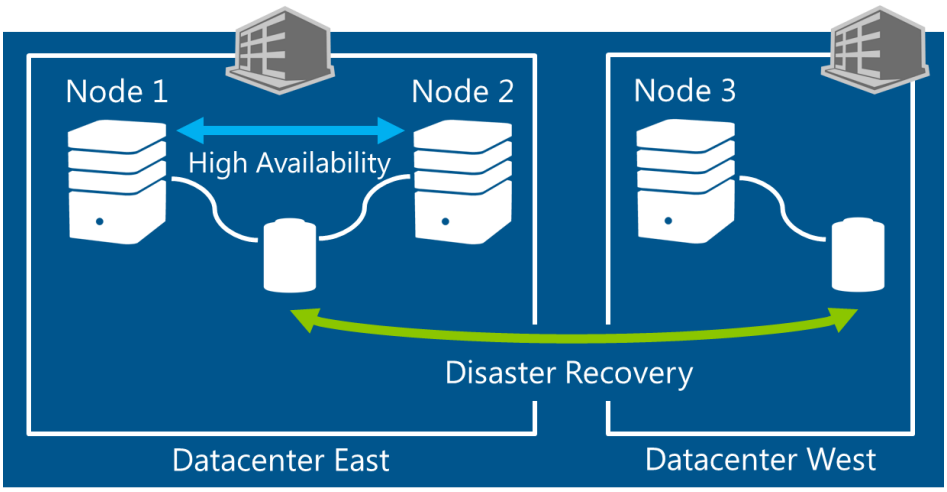

***Figure 1**: High availability protects against a single server failure. Disaster recovery protects against data center failure.*

What Hosted SQL Server with AlwaysOn Offers Customers

SQL Server 2012 AlwaysOn solutions provide redundancy across servers and across data centers. Planned and unplanned downtimes are minimized. The administrator has new client tools to make deploying, monitoring, and managing HA and DR solutions easier.

SQL Server 2012 AlwaysOn solutions include:

- AlwaysOn Failover Cluster Instances
- AlwaysOn Availability Groups
- Availability Group Listeners
- Read-only Secondary Replicas and Application Intent

Using these solutions, hosters can offer their customers a variety of HA and DR designs.

AlwaysOn Failover Cluster Instances can be used for both HA and DR. Entire SQL Server instances fail over from one node to another. Flexible rules can be implemented that determine when the primary node is not healthy and when automatic failover should occur. For example, secondary nodes detect when the primary node has failed. Rules can restrict votes to nodes within a data center, so a network failure does not trigger a failover.

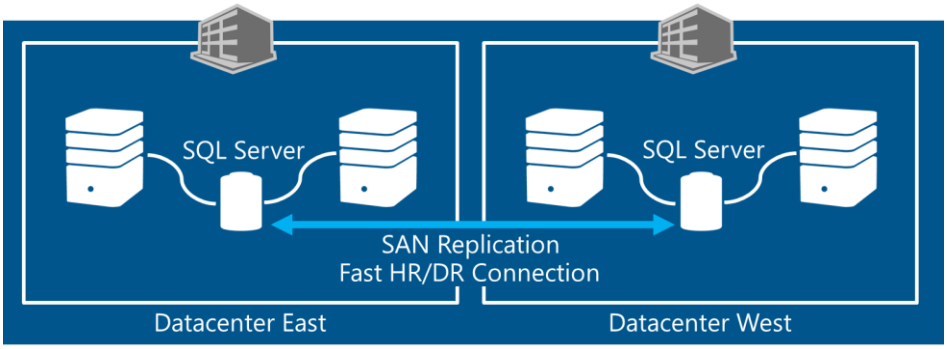

Figure 2: *High availability with Failover Cluster Instances across distances requires SAN replication and a fast network connection between storage entities.*

Windows Server® 2012 Standard Edition offers two-node clusters. Failover clusters required the Enterprise or Datacenter versions of Windows Server prior to 2012. This change offers immediate cost benefits to the hoster offering HA SQL Server solutions. The Enterprise Edition offers as many as 64 nodes to guarantee reliability to the customer's enterprise applications running in the cloud.

A Hyper-V® cluster with highly available virtual machines has a limit of 1024 VMs per node and 8000 VMs per cluster, increasing the maximum capacity of a host server.

Hyper-V virtual machines can start with a priority setting. Clustered virtual machines with a higher priority start before those with a lower priority during a failover. If lower priority virtual machines are starving a cluster of memory or other resources, they are taken offline until higher priority VMs can start. A hoster can prioritize virtual machines according to price point and the importance of an application.

SQL Server 2012 can make use of Hyper-V Dynamic Memory that can dynamically allocate memory among running VMs. The amount of memory available to a VM changes with demand. This allows SQL Server instances that are highly active to be allocated more resources from the physical server, while VMs that are inactive can give up memory.

Local failover clusters provide HA within a data center, protecting against server failure. Nodes are directly connected to shared storage or share a SAN. Microsoft also offers the iSCSI Software Target over the network by turning a Windows Server computer into a storage device. This can be used as shared storage for a failover cluster, reducing the expense for the hoster versus the setup costs of a SAN.

With multisite clustering of a SQL Server instance, cluster nodes can be located on different subnets or use virtual LANs. Failover can now be across distances, with more flexible failover policies. Low-level infrastructure such as SAN replication and a sufficiently fast connection are required to keep storage entities in sync. This takes clustering beyond the HA role and into the DR role with failover now protecting against data center failure.

AlwaysOn Availability Groups can also be used for both HA and DR by maintaining a set of databases, synchronously or asynchronously, in different locations. The databases are not hardware-dependent, so the hoster has the flexibility of using a wider variety of hardware and can make use of more available resources. Availability groups roll database mirroring and log shipping into a single solution, reducing maintenance overhead for the administrator. AlwaysOn HA and DR features are available starting in the SQL Server 2012 Enterprise Edition.

Note: Database mirroring was deprecated as of SQL Server 2012 and will be removed from a future version of SQL Server. Availability groups are intended to replace database mirroring starting with SQL Server 2012.

Each availability group contains a defined set of databases known as replicas. A single primary replica writes to a set of up to four secondary replicas. Writes are synchronous or asynchronous. Synchronous writes can occur on up to two secondary replicas. A synchronous write means that a transaction does not commit on the primary replica until its success is guaranteed on all secondary replicas where synchronous writes are specified. An asynchronous write takes place some milliseconds after a transaction commits.

If a server or a data center is lost, automatic failover can occur to one secondary replica where synchronous writes occur. Manual failover can occur to any replica.

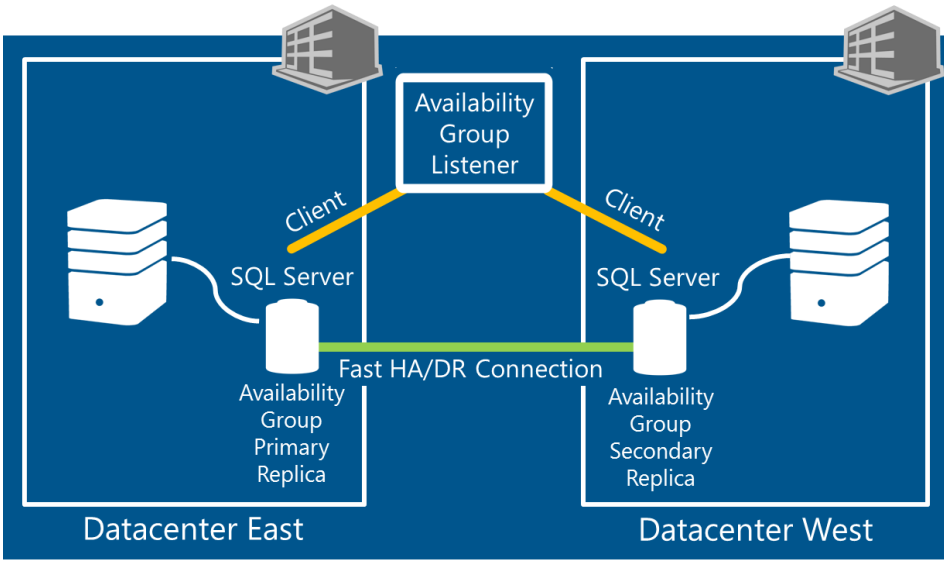

Figure 3: *Availability groups are sets of databases that can be HA with synchronous writes and a fast connection; DR with databases in different data centers.*

If a failure occurs on the primary replica before the write occurs on an asynchronous secondary replica, there can be some data loss, just as with asynchronous database mirroring or log shipping. The advantage to taking the small risk inherent in asynchronous commits is that the primary replica is not waiting for writes to these servers and can release locks and process other reads and writes faster. This is especially helpful where a fast connection does not exist between the primary replica and a secondary.

Availability group listeners are a network end point for an availability group. They allow clients to connect to the primary replica with a single network name, IP address, and TCP port. A listener also connects to a secondary replica when configured for read-only secondary replicas. When a failover occurs, there is no need for the client to know the network name of the secondary replica that is the new primary replica. Listeners fail over as the primary replica fails over.

Read-Only Secondary Replicas and Application Intent allow clients to use secondary replicas for read-only purposes, such as reporting and backups. A client can connect to a read-only secondary replica by specifying an application intent attribute in a Microsoft .NET connection string when connecting to the listener. The listener routes the client connection to an appropriate secondary replica.

The Hoster Designs Flexible Solutions

With these features, a hoster has more flexibility in designing solutions that allow the customer to choose more cloud offerings.

- Traditional failover cluster instance. Cluster nodes are in a single data center using shared direct-access switch connection storage or a SAN. Clusters may fail over SQL Server instances or virtual machine hosts.
- Failover across data centers. Failover cluster instances can have nodes in different data centers with asymmetric storage, i.e., storage is visible only to nodes in a single data center, assuming sufficiently fast connections. Cluster nodes can be on different subnets. Again, the cluster is failing over SQL Server instances or virtual machine hosts.
- On premise-off premise availability groups. The hoster can offer the customer an offsite data center as a secondary replica in an availability group. The customer might still have their primary active location be within their own data center. An availability group destination offsite gives the customer a relatively inexpensive disaster recovery solution. The customer doesn't need to invest in infrastructure, and the hoster can mix and match hardware to the customer's needs by isolating a set of databases without the strict requirements of a true failover cluster instance.
- Availability groups for disaster recovery. The hoster can offer the customer availability groups across data centers without the fastest possible connection between primary and secondary replicas. Writes to the secondary replica are asynchronous, and during failover some data may be lost; but that may be an acceptable risk for the customer to have a less expensive option for keeping the application running offsite.

AlwaysOn and Windows Server Failover Clustering

AlwaysOn Failover Cluster Instances and AlwaysOn Availability Groups use the Windows Server Failover Clustering (WSFC) infrastructure. Both AlwaysOn components are registered as cluster resources.

Differences between Failover Cluster Instances and Availability Groups

Failover cluster instances fail over an entire SQL Server instance with the new active node taking control of storage containing all database files for both system and user databases. Availability groups are managed at the SQL Server level and consist of a set of databases.

The failover cluster instance uses shared storage for high availability within the data center; or, asymmetric storage, shared between a subset of cluster nodes, across data centers for both high availability and disaster recovery.

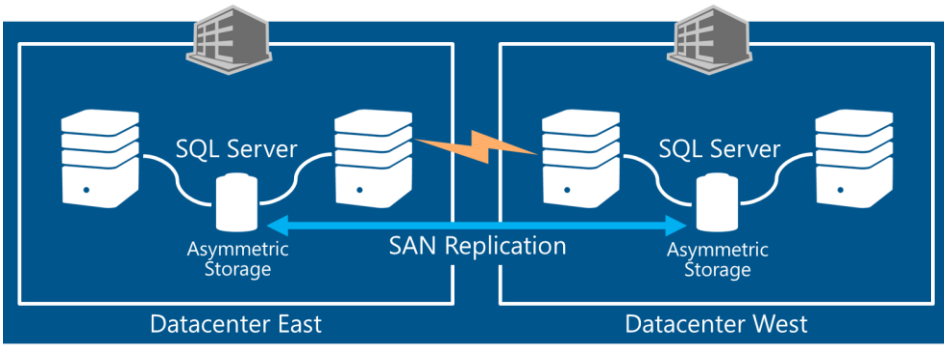

Figure 4: *With asymmetric storage, nodes see one set of storage but not another in HA across data centers.*

Availability groups also use asymmetric storage within or across data centers. The storage on one node is not visible to other nodes. This allows nodes within an availability group to use any type of storage that meets the needs of the SQL Server databases being hosted on the cluster node. Because failover is among a set of databases, the availability group does not have the same requirements for matching configurations at the server level.

The Validate a Configuration test will fail a storage test when setting up different non-shared storage types for different nodes. You can choose to bypass the storage test and still pass the cluster validation test.

The Windows Server Failover Clustering infrastructure provides the availability group the single network name, IP address, and TCP port used by client connections, detecting when a SQL Server instance is no longer available according to quorum configuration rules, and accomplishing the failover.

Because both solutions use the underlying Windows Server Failover Clustering infrastructure, the following planning and deployment considerations apply to both failover cluster instances and availability groups.

Hardware

Hardware must be certified for use with the intended version of Windows Server. For more information, see Failover Clustering Hardware Requirements and Storage Options.

It is a best practice to have redundant networks, switches, routers, and network adapters in order to avoid single points of failure.

Storage for a failover cluster instance across distance may be asymmetric, i.e., storage for one node may not be visible to another node, whether within or across data centers. An availability group replica may have a single node connected to storage, even though it is part of a cluster.

There is a hotfix for Windows Server 2008 and Windows Server 2008 R2 that allows the Failover Cluster Manager snap-in for Microsoft Management Console to support asymmetric storage.

After the hardware is configured, you can validate the hardware for a Windows Server 2012 failover cluster by running the Validate a Configuration Wizard. This is also available as the Test-Cluster Windows PowerShell™ cmdlet. These test the servers, network, and storage to validate that the configuration can support Windows Server Failover Clustering.

When setting up an availability group with asymmetric storage of different types on different nodes, the storage test portion of the Validate a Configuration may fail. You can choose not to test the storage, and the cluster can still pass the validation test and be a fully supported solution.

For more information, see:

- Validate Hardware for a Windows Server 2012 Failover Cluster;
- Failover Cluster Step-by-Step Guide: Validating Hardware for a Failover Cluster (Windows Server 2008 R2).

Cluster Services Installation

Windows Server Failover Clustering requires the Enterprise Edition of Windows Server 2008. The Standard Edition of Windows Server 2012 supports two node clusters; the Enterprise Edition is required for more than two nodes. SQL Server 2012 is supported on Windows Server Core.

Each computer must be running the x86 or x64 versions. WOW64 is not supported for SQL Server Failover Cluster Instances.

Each node in the cluster must be running the same version of Windows Server at the same patch level. The SQL Server versions and hot fixes must be the same.

Antivirus software may cause problems with cluster services and should not be installed on any node.

Only users that are, at a minimum, members of Local Administrators on each cluster node should install and configure Windows Server Failover Clustering. Being a member of Domain Admins is helpful. If the administrator is not a member of Domain Admins, that user account needs Create Computer Objects and Read All Properties permissions in the Organizational Unit.

Administrators can use the Server Manager and Failover Cluster Manager utilities to install cluster services, tools, and to manage clusters using a Windows user interface, or using Windows PowerShell cmdlets. Server Manager can manage ad-hoc groups of servers, such as all the servers in a domain or in a data center.

Server Manager has been redesigned for Windows Server 2012, and this version is not fully capable of performing all the same functions on Windows Server 2008 or Windows Server 2008 R2.

For more information, see Installing the Failover Cluster Feature and Tools in Windows Server 2012.

Failover Cluster Manager is one of the tools that is installed with cluster services. After cluster services are installed, Failover Cluster Manager can be used to create a cluster, add nodes, and manage the cluster.

For more information, see Creating a Windows Server 2012 Failover Cluster.

Network Configuration

All nodes of a failover cluster instance must be in the same domain. Every server should have the same domain role, that of a member server. SQL Server Failover Cluster Instances are not supported on domain controllers.

All cluster nodes should use the same domain account to log on to the network. Likewise, each SQL Server instance should use the same domain account.

Network names and IP addresses are assigned to the following resources. Network names must be unique in the domain. SQL Server 2012 supports IPv6.

- The name of the SQL Server Failover Cluster Instance. This is the virtual server name which clients will use to connect to SQL Server. It is not the same thing as a SQL Server named instance (see below).
- The name of each cluster node.
- The name of the failover cluster.

If connecting to a default instance of SQL Server, the client does not have to supply a named instance name. When connecting to a named instance, the client must furnish the named instance name as part of the connection string along with the network name or IP address assigned to the SQL Server Failover Cluster Instance or SQL Server Availability Group. The administrator assigns the instance name during SQL Server setup. Instance names do not have to be unique in the domain, only to the server hosting the SQL Server instance. In the case of a cluster, the named instance name is assigned to all failover cluster instances on all nodes.

Administrators should use caution when assigning SQL Server named instance names. To change one requires uninstalling and reinstalling SQL Server.

An instance ID is also supplied during SQL Server setup. This defaults to MSSQLSERVER for the default instance or the named instance. The instance ID is used to identify the instance of SQL Server in a directory tree or in the registry. Like the named instance name, the instance ID cannot be changed without reinstalling SQL Server.

You can also assign a static TCP port for the SQL Server instance that clients will use to connect. SQL Server uses port 1433 by default. If there is more than one instance of SQL Server using the same network name and IP address, one should either be assigned a different static port, or SQL Server can assign a dynamic port. Clients can connect to the server using UDP port 1434 to get the TCP port they should use for the specified instance from the SQL Server Browser service. An administrator would typically assign the default port 1433 to the default instance and use a different static port or dynamic port assignment with named instances.

The SQL Server Browser service is not a cluster resource and does not fail over. It must run on each node of the cluster. On clusters, the service listens on IP_ANY.

For more information, see SQL Server Browser Service.

SQL Server 2012 supports nodes for a single failover cluster instance in different subnets. As with previous versions of SQL Server, you can also create a Stretch Virtual LAN (VLAN) to expose a single IP address.

In previous versions of SQL Server, during startup SQL Server iterates through all IP addresses assigned to the SQL Server Failover Cluster Instance and attempts to bind all of them. If any binding fails, the SQL Server startup fails. Starting with SQL Server 2012, when adding a node, SQL Server Setup detects a multi-subnet installation and specifies an IP address resource dependency of OR so that a SQL Server can be online when there is at least one IP address it can bind to.

For more information, see:

- SQL Server Multi-Subnet Clustering;
- SQL Server 2012 AlwaysOn: Multisite Failover Cluster Instance.

Quorum Configuration

Quorum configuration is a necessary part of cluster configuration. For both failover cluster instances and availability groups, the health of the cluster node and the SQL Server services are constantly monitored by the other nodes.

Part of installation and configuration of a cluster will be to establish the quorum rules for when an automatic failover should occur, at the cluster node level, the SQL Server FCI level, and the availability group level.

If a node or the SQL Server instance on a node is unresponsive, the node is considered to have failed. A node is considered healthy only if a quorum of other nodes in the cluster votes to consider it healthy. Part of the quorum configuration is to establish the methodology for voting and what constitutes failure.

For example, if all nodes are in a single data center using shared storage, all nodes by default all nodes have a vote. If the cluster is configured across data centers, an administrator may wish to configure the cluster so that nodes in a separate data center do not have a vote. This will prevent a network failure across data centers from triggering a failover.

For availability groups using asymmetric storage, Microsoft recommends the Node Majority quorum mode when there is an odd number of voting nodes, and the File Share Majority mode when there is an even number of voting nodes.

For more information, see:

- WSFC Quorum Modes and Voting Configuration;
- Configure and Manage the Quorum in a Windows Server 2012 Failover Cluster;
- Failover Policy for Failover Cluster Instances.

SQL Server Setup

After the cluster services installation is complete, and before installing SQL Server, install the prerequisites on each node. Windows PowerShell is not installed by SQL Server 2012, and Windows PowerShell 2.0 is required. Installing .NET 4.0 may require a reboot.

For more information, see:

- What's New in Failover Clustering (Windows Server 2012);
- Windows Server Failover Clustering with SQL Server.

If you are installing SQL Server in a Hyper-V environment, also see Support policy for Microsoft SQL Server products that are running in a hardware virtualization environment.

A SQL Server Failover Cluster Instance runs in a WSFC resource group. A resource group is a collection of resources, like the SQL Server services, that are monitored and managed as a single unit. Each node in a resource group syncs configuration and registry settings. The same cluster can have more than one failover cluster instance resource group; however, the same instance of SQL Server cannot be a member of more than one resource group.

The Distributed Transaction Coordinator (MSDTC) service can be part of a SQL Server Failover Cluster Instance resource group. Before installing SQL Server, you should determine if the MSDTC service is required. If you are installing the database engine only, MSDTC is not required. If you are installing the database engine and Integration Services, workstation components, or will use distributed transactions, MSDTC should be part of the resource group. For more information, see Before Installing Failover Clustering.

SQL Server instance names must be different within the same failover cluster instance.

The SQL Server setup detects the existence of the cluster, and you can add nodes through setup at the time of installation. The SQL Server version, service packs, and hot fixes must be identical on all nodes.

Services start and stop on each node through WSFC. Only one node in the resource group is the primary, active node at any given time.

For more information, see AlwaysOn Failover Cluster Instances.

After a Failover Cluster Instance is installed, it can be changed, repaired using the SQL Server setup program. You can add nodes to an existing resource group, remove a node, or change a SQL Server instance to a stand-alone installation.

The service accounts the SQL Server services use to log on to the network should be consistent across all nodes. Passwords should not change unless you can change them on all nodes immediately. Changes to service accounts should be made through the SQL Server Configuration Manager.

If you intend to use Kerberos authentication for client connections to the failover cluster instance, a Service Principal Name (SPN) must be registered with Active Directory. The SPN maps to the service account that the SQL Server service uses to log on to the network.

For more information, see Register a Service Principal Name for Kerberos Connections.

Migrating to Failover Clustering and Availability Groups

One of the biggest challenges for a hoster may be migrating existing SQL Server instances from legacy configurations to new SQL Server 2012 Failover Cluster Instances and Availability Groups. Reconfiguration is necessary where failover clustering has been used only for local high availability and database mirroring for disaster recovery.

Many older solutions are in place that have a Windows Server cluster in a data center providing local high availability with database mirroring, log shipping, or replication being used to copy SQL Server data to remote data centers for disaster recovery. In the secondary data center, there may be another local cluster in place providing local high availability for the cluster nodes hosting the SQL Server instance on the receiving side of the mirroring, log shipping, or replication data.

Availability groups require all nodes be a part of a single Windows Server Failover Cluster. The storage for the cluster nodes can be asymmetric between data centers, allowing the clusters to fail over locally for high availability and remotely for disaster recovery. But because different clusters need to be reconfigured to a single Windows Server Failover Cluster, the migration path can be complex.

There are typically two migration paths from legacy solutions to AlwaysOn solutions. One migration path uses intermediate hardware, and one does not. They each have advantages and disadvantages.

Migrating without intermediate hardware is simpler, but there will be times during the process where there will not be data redundancy or protection.

Migrating with intermediate hardware is more complex, but preserves high availability and disaster recovery throughout the process.

In either case, clients will need to be reconfigured during the migration.

For more information, see Migration Guide: Migrating to SQL Server 2012 Failover Clustering and Availability Groups from Prior Clustering and Mirroring Deployments.

AlwaysOn Failover Cluster Instances

A failover cluster instance is a single instance of SQL Server installed across Windows Server Failover Clustering nodes. With SQL Server 2012, instances can be installed on nodes in multiple subnets. On the network, there is only one network name, IP address, and port that follow the active instance from node to node after a failover.

There is only one SQL Server instance that is considered the resource group owner. No other node in the failover cluster instance is running its SQL Server services.

Failover

When a server fails, due to hardware or operating system failure, or an application like SQL Server fails, other nodes detect the absence of the server and/or the clustered applications it hosts, and fail it over to another node according to a flexible quorum configuration. During an upgrade or migration, an administrator may fail over manually.

The SQL Server running on the old primary node does a checkpoint in all databases, if it is still running. This process writes dirty pages in cache to disk. This does not affect whether a transaction is successful or not, but it does affect how much time the new node will take to recover each database during startup.

All the SQL Server services on the old primary server are stopped.

The new primary node takes ownership of the shared storage where clustered application files reside. The SQL Server services and other clustered application services start automatically.

The virtual network name and IP address are transferred to the new node.

A SQL Server Failover Cluster Instance resource group unit of failover includes:

- The SQL Server service (the database engine).
- The SQL Server Agent service.
- The Distributed Transaction Coordinator service, if installed.
- The Analysis Server service, if installed.
- One file share resource, if FILESTREAM is installed.

SQL Server reads the transaction logs of the databases that just failed over, and rolls back transactions that were in progress at the time of the failover. Transactions that were committed only in the transaction log and did not have their data pages successfully written to disk as part of the checkpoint on the old active node are rolled forward.

Depending on how many dirty pages there are in cache at the time of the failover, the checkpoint on the old active primary or the roll forward on the new primary node can cause the failover to take a long and unpredictable length of time. Starting in SQL Server 2012, a SQL Server database engine failover encompasses the entire SQL Server instance, including all system databases, with the possible exception of the Tempdb database, and user databases.

All server and database metadata, such as logins, server roles, which logins map to which database users, scheduled jobs, and job history, all fail over as a unit. This makes administration easier for a hoster offering Infrastructure as a Service (IaaS) or a customer that might have multiple databases on a server, such as an extract, transform, and load (ETL) staging database and a data warehouse.

The client will experience a period of down time as the failover, roll forward, and rollback processes complete on the new active node. When the new active node is available, a client can reconnect using the same single network name and port. The network name will resolve to the same IP address.

A SQL Server 2012 cluster built across subnets no longer needs a stretch virtual LAN as with previous versions of SQL Server, however, that can still be used if desired. With the introduction of the OR dependency, nodes can use one IP address OR another IP address that may be in a different subnet, and the cluster is still considered to be online serving clients.

When a SQL Server client reconnects after a failover, logic should be built into a client and into the database design to detect and retry transactions that were rolled back at the time of the failover.

Storage

Storage on a failover cluster instance is directly connected to nodes or accessible by network. Shared storage, either directly connected or on a SAN, is used by a failover cluster instance to host the files that will fail over as part of the application failover. With SQL Server, this includes all user database files that are part of a clustered instance.

In SQL Server 2012, the Tempdb database files are not required to be on shared storage; they can be on storage local to individual nodes, but the path to Tempdb database files must be the same on all nodes.

Each cluster should have a separate set of physical disks configured at the hardware level according to the desired RAID specification. Different clusters cannot share the same set of physical disks.

SQL Server 2012 and WSFC support iSCSI storage devices. If you are deploying SQL Server on Hyper-V VMs, this is as easy as setting up another VM for storage. This makes the creation of test environments easier.

Each clustered server using iSCSI should have one or more network adapters or host bus adapters that are dedicated to storage. A separate network should be dedicated to iSCSI and not be used for other network communication.

For more information, see Quickly Setup SQL Server AlwaysOn Failover Cluster Instance in Hyper-V.

Starting with SQL Server 2012, database files can reside on SMB file shares. This can help partition administration by doing away with the 26 drive letter limitation.

Cluster Shared Volumes do not support the typical SQL Server 2012 workload on a cluster node. For more information, see Use Cluster Shared Volumes in a Windows Server 2012 Failover Cluster.

For failover cluster instances with nodes in different data centers, SAN replication and failover technology is required. For more information, see SQL Server 2012 AlwaysOn: Multisite Failover Cluster Instance.

For more information on design and implementation of failover cluster instances within and across data centers, see AlwaysOn Architecture Guide: Building a High Availability and Disaster Recovery Solution by Using Failover Cluster Instances and Availability Groups.

AlwaysOn Availability Groups

AlwaysOn Availability Groups are new with SQL Server 2012. Using the WSFC infrastructure, availability groups provide an HA and DR solution to fail over a set of databases, potentially over long distances. Availability groups are available in the Enterprise edition of SQL Server 2012.

Note: Availability groups are intended to replace Database Mirroring. Database mirroring has been deprecated and will be removed from a future version of SQL Server.

Availability Groups are defined in terms of replicas. Primary or secondary replicas are hosted on different SQL Servers which are on a failover cluster instance.

There is a single primary replica where all writes occur.

There can be up to a total of four secondary replicas. All secondary replicas are read-only.

Transactions in a Database

The architecture of SQL Server consists of 8K pages. Both persistent storage in database files and most of SQL Server's memory are made up of these pages.

When a Transact-SQL command searches for a row of data, SQL Server scans an index or a table. If the page where the row resides is not already in memory, SQL Server loads it into memory from a database file.

When a transaction adds, changes, or removes rows from a table, these changes are made to these table and index pages and written to the database's transaction log. In order to maximize performance, entire rows are not written back to disk immediately. Data and index pages are cached and are only written to the database files the next time SQL Server does a checkpoint in the database. SQL Server writes the changes made to the rows to the transaction log, and these are written to persistent storage when the transaction commits. Until the transaction commits, it can be rolled back. After the transaction commits, the changes are made and there is no going back except by starting another transaction.

When a database is part of an availability group, when the transaction commits and SQL Server writes the changes to persistent storage on the primary replica, those changes are also made to the transaction log of the secondary database, and the table and index pages in the secondary replica change.

While these writes on the secondary replicas are taking place, the transaction on the primary replica is either waiting for the secondary replicas to finish and signal success for its commits, or the primary replica has already committed and moved on.

Synchronous Commit Mode

Two of the four secondary replicas can operate in synchronous commit mode. This is an HA and/or DR solution at the replica level, given a suitably fast network connection between the primary and secondary replicas.

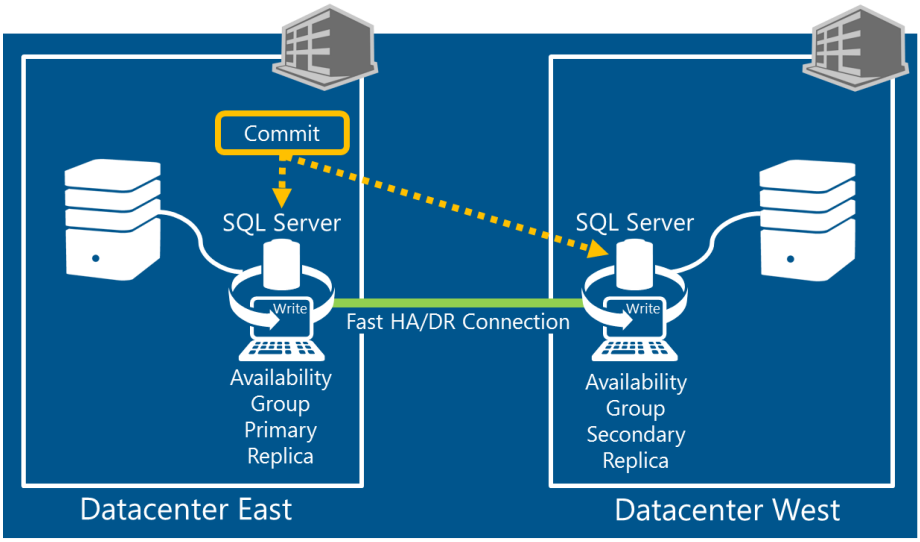

Figure 5: *Transactions commit on all synchronous commit replicas simultaneously. A fast connection is required to prevent latency on the primary replica.*

In synchronous commit mode, a transaction will not commit in a primary replica database until writes to all secondary synchronous commit replicas are successful. If there is increased transaction latency, that also means that write locks in the primary replica database can potentially be held longer, which, in turn, means greater potential for perceptible delays and increased resource usage.

For automatic failover to occur, both the primary replica and the target secondary replica must be set to synchronous commit mode.

Asynchronous Commit Mode

Other secondary replicas, up to the four total allowed, operate in asynchronous commit mode. A transaction can commit in the primary replica before a write takes place in secondary asynchronous commit replicas. There is no additional transaction latency, and the client is notified of transaction success immediately. This is a DR solution when the secondary replicas are in different locations.

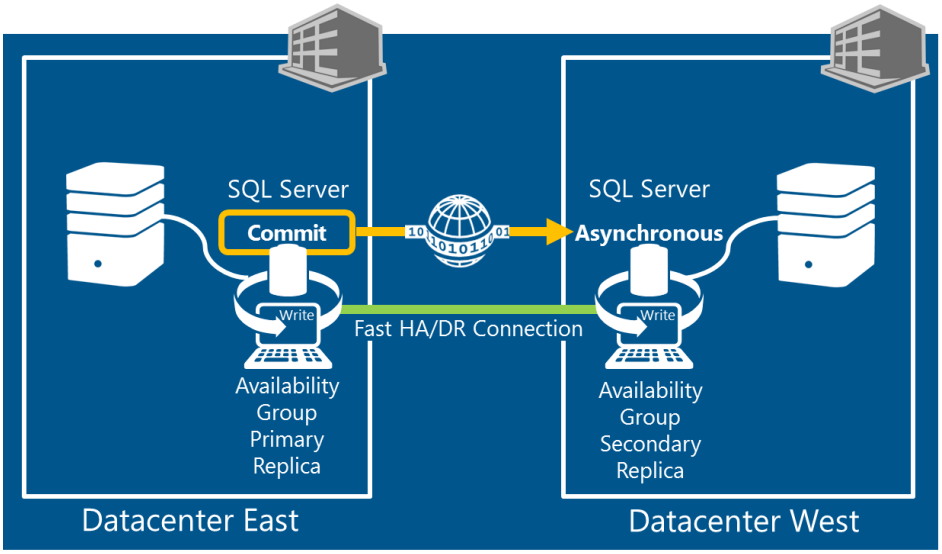

Figure 6: *Transactions commit on the primary replica without waiting for secondary replicas set to asynchronous commit. There is no latency on the primary replica as a result. Only manual failovers are permitted to secondary replica, and there may be data loss.*

The slight risk with asynchronous commit mode, as with any asynchronous DR solution, is that a failure occurs after a transaction commits and before it has a chance to be written to the secondary replicas. Some data may be lost as there is no guarantee that a secondary replica will be in sync with the primary replica. Writes to the secondary replicas can lag behind at any point. Errors on a secondary replica do not affect the primary replica.

If the primary replica is set to asynchronous commit mode, all secondary replicas must also receive data in asynchronous commit mode.

For more information, see Availability Modes (AlwaysOn Availability Groups).

Failover

One of the benefits of synchronous commit mode is that an availability group can fail over automatically with no data loss when the primary node is detected as having failed. You can configure automatic failover from the primary replica to a single secondary replica that uses synchronous commit mode.

Administrators can also manually fail over from the primary replica to any secondary replica.

The set of databases making up the availability group is all that fails over. System databases are not part of the availability group. Logins, server roles, and server-level permissions do not fail over unless you are using Contained Databases (see below).

WSFC and Availability Groups

Each instance of SQL Server must reside on a different node of a single Windows Server Failover Cluster. The cluster should be installed and verified, and a failover should be rehearsed.

A SQL Server instance hosting an availability group can also be part of a failover cluster instance, or it can be a stand-alone instance. A stand-alone instance of SQL Server hosting a replica is still considered a cluster node, but an entire instance will not automatically fail over to or from this node. Only manual failovers can occur at the cluster instance level when the cluster hosts an Availability Group. The Availability Group can still fail over automatically at the database level if the primary replica and a secondary replica are in synchronous commit mode.

A Windows Server Failover Cluster that hosts an availability group behaves differently than when it hosts a failover cluster instance. A failover cluster instance will fail over entire cluster resource groups, including SQL Server instances. The cluster services hosting an availability group detect a node has failed and trigger an automatic failover of the availability group only. A failover cluster instance hosting an availability group can only fail over the cluster resource groups manually.

WSFC monitors the health of the availability group. A WSFC Resource Group is created for every availability group.

The health of the availability group depends on the votes of a quorum of nodes in the cluster. All nodes have a vote even if a node by default or even if a node does not host a replica.

Special care should be taken in configuring a quorum model when the WSFC has nodes in different data centers. You may, for example, configure nodes in the primary data center to have votes as to whether the primary replica of an availability group is in good health or not, but not allow nodes in the secondary disaster recovery data center to have any votes. This will prevent a failover being triggered by a network outage between the two data centers.

A hotfix is available to configure a node as non-voting in Windows Server 2008 R2.

For more information, see WSFC Quorum Modes and Voting Configuration.

SQL Server should be installed and working on all nodes. A SQL Server hosting a secondary replica can actively host other databases as well, unless a node is a secondary node in a failover cluster instance.

Each SQL Server instance must use the same SQL Server collation.

All prerequisites should be installed. Note that for clients to use the new connection string attributes to connect to a secondary replica database, there is a hotfix for .NET 3.5 SP1 to be installed on all clients.

All servers hosting SQL Server instances in the availability group must be in the same domain. All SQL Server service accounts must be the same for all nodes.

If SQL Server uses Kerberos authentication, a Service Principal Name must be registered that maps to the domain account SQL Server is using to log on to the network. This is not required for NTLM authentication.

For more information, see Prerequisites, Restrictions, and Recommendations for AlwaysOn Availability Groups.

Enable Availability Groups on Each SQL Server Instance

Enable availability groups one node at a time through the SQL Server Configuration Manager or Windows PowerShell. Enabling Availability Groups may cause the SQL Server to require a restart. For more information, see Enable and Disable AlwaysOn Availability Groups.

Create Endpoints for SQL Server to SQL Server Connections

Each server in the availability group must have a dedicated DATABASE_MIRRORING endpoint to receive connections from other server instances. This endpoint must be created before creating the availability group. This endpoint is used exclusively to receive connections from other SQL Server instances. Endpoints listen on a unique TCP port.

If the network between SQL Server instances in an availability group has a firewall between instances, this endpoint port must be allowed both incoming and outgoing connections.

These SQL Server instances are part of failover cluster instances and should be logging on to the network using domain user accounts. In this case, Windows® authentication can be used for the endpoints. If all the SQL Server instances in the availability group are using the same domain user account, the correct user logins should exist automatically on each server instance. You can verify this by using Object Explorer in SQL Server Management Studio. This simplifies the security configuration and is recommended.

By default, an endpoint requires encryption be enabled for data sent over availability group connections. Unless you can guarantee that your network is secure, it is a best practice to create the endpoint with encryption.

When specifying the encryption algorithm, please note that the RC4 algorithm is deprecated and will be removed from a future version of SQL Server. The AES algorithm is recommended.

The New Availability Group Wizard and the Add Replica to Availability Group Wizard can create the endpoint and grant the necessary permissions to the SQL Server instance service account.

For more information, see:

- The Database Mirroring Endpoint;
- Transport Security for Database Mirroring and AlwaysOn Availability Groups;
- Specify the Endpoint URL When Adding or Modifying an Availability Replica;
- CREATE ENDPOINT (Transact-SQL).

Configuring an Availability Group for FILESTREAM

The FILESTREAM feature allows SQL Server databases to store unstructured BLOB data in the file system independent of other database files. If any databases using FILESTREAM will be part of an availability group replica, enable FILESTREAM on every instance of SQL Server hosting a replica.

For more information, see:

- Enable and Configure FILESTREAM;
- FILESTREAM (SQL Server).

Create an AlwaysOn Availability Group

At this point in the process, cluster services for a single WSFC cluster are installed on each node where the availability group will reside. This is not the same thing as a failover cluster instance, as an Availability Group node may contain only a stand-alone SQL Server.

SQL Server is installed on a different node of the same WSFC cluster.

You can create an availability group at a minimum on the primary replica. You can specify up to four additional secondary replicas at the same time, or you can add them later.

To create and configure AlwaysOn Availability Groups, you have the choice of using:

- The New Availability Group Wizard;
- Windows PowerShell cmdlets;
- The Transact-SQL CREATE AVAILABILITY GROUP command.

You should be prepared to supply the following information, whichever method you choose:

- The name of the availability group. This must be unique.
- The names of the databases to be included in the replica.
- Where a backup job should perform backups, given the role of the replica. You can specify an automated backup preference of:
    - Primary replica only.
    - Secondary only. If the primary replica is the only one online, backups do not run.
    - Secondary replicas unless the primary replica is the only one online.
    - None. The role of the replica is ignored when choosing where to perform backups.
- The Failure Condition level. There are five choices for conditions under which a failover may be triggered. For more information, see Configure the Flexible Failover Policy to Control Conditions for Automatic Failover.

- The Health Check Timeout in milliseconds. This value is how long a SQL Server health diagnostic stored procedure should wait before assuming the replica has failed. It applies only to those replicas that have an Availability Mode of Synchronous Commit with automatic failover.
- From one to five replicas, including the primary replica and up to four secondary replicas, with the following specified for each:
    - Each replica is identified by the network name of the SQL Server and an instance name.
    - The endpoint URL for the replica.
    - The Availability Mode, either Synchronous Commit or Asynchronous Commit. Synchronous Commit may be specified for up to three replicas, including the primary. Asynchronous Commit may be specified for up to five replicas, including the primary. For more information, see Availability Modes.
    - The failover mode for the replica, either automatic or manual. Automatic failover is allowed for the primary replica plus only one secondary replica. Both replicas must have Synchronous Commit set for the Availability Mode.
    - A priority for performing backups on this replica. Values range from 0 to 100, with 0 meaning never do a backup on this replica, 1 meaning only do a backup on this replica if no other replica is available, and 100 meaning the highest priority.
    - The role of the replica, either primary or secondary.
    - For secondary replicas, whether connections are allowed to secondary replica databases. If connections are allowed, they can only be read-only. For more information, see Active Secondaries: Readable Secondary Replicas.
    - For the primary replica, whether read-only connections are allowed or only read-write.
    - For secondary replicas, you can optionally supply a read-only routing URL. For the primary replica, you can optionally supply a list of secondary SQL Servers that will accept read-only connections. See Read-Only Secondary Replicas and Application Intent, below.
    - An optional Session Timeout value, in seconds. At least 10 seconds is recommended.
- An Availability Group Listener IP address and port. Having an Availability Group Listener is recommended, though the New Availability Group Wizard does not create one by default. See Availability Group Listeners, below.

Changing an Availability Group

New secondary replicas can join an existing Availability Group using SQL Server Management Studio Wizards, Windows PowerShell cmdlets, or Transact-SQL commands.

The first step is to add the secondary replica. Of the following properties listed previously for adding a new Availability Group, the following are required for adding a secondary replica.

- The network name of the SQL Server instance and an instance name.
- Endpoint URL.
- Availability Mode.
- Failover Mode.

For more information, see Add a Secondary Replica to an Availability Group.

Adding the replica to the availability group manually, using Transact-SQL or Windows PowerShell, does not start any data synchronization movement. You may be able to do the initial synchronization using Wizards to add a replica or a database (see below).

After the secondary replica is added to the availability group, connect to the new secondary replica and join the new replica to the availability group. In order for this join operation to succeed, the primary replica must be online and the new secondary server must be able to connect to the database mirroring endpoint of the SQL Server instance hosting primary replica.

For more information, see:

- Join a Secondary Replica to an Availability Group;
- Administration of an Availability Group.

Initial Synchronization of Primary and Secondary Replicas

After a secondary replica is added to the availability group, you must synchronize the availability group databases with the primary replica. Likewise, adding a new database to an existing availability group, the data synchronization process is the same.

Synchronizing databases can be done manually or by using one of the wizards that are used to manage replicas.

Before starting the initial synchronization process, stop transaction log backups from the primary replica availability group databases. Do not resume log backups until the secondary replica has been prepared and synchronized with the primary replica.

If you use the Create New Availability Group Wizard, the Add Replica to Availability Group Wizard, or the Add Database to Availability Group Wizard, the AlwaysOn Select Initial Data Synchronization page is used to specify a preference for initial synchronization of secondary databases. You may choose:

- Full synchronization. This does a full backup and restore from the primary replica to the secondary replica(s) and starts data synchronization automatically. In order for this option to be used:

- The database files all use the same paths and file names on each server.
- The database names do not already exist on any secondary replica.
- You will need to specify a network file share with enough storage space available to store the database backup file.
- The domain user account used by the primary replica SQL Server instance must have read and write permissions at the backup network share.
- The domain user account used by the secondary replica(s) SQL Server instance must have read permissions at the backup network share.

NOTE: Make allowances for how much time will be required for the transfer of very large amounts of data across the network, from the primary replica to the backup network share, then from the share to the secondary replicas.

- Join Only synchronization. The databases already exist on the secondary replica(s), and you have already prepared each secondary database (see below).
- Skip initial data synchronization. You intend to prepare each database later (see below).

To prepare a secondary database means to manually make a full database backup of the primary replica databases and to restore them using the WITH NORECOVERY option to each secondary replica. If database file paths or names must be different on a secondary replica, you must also use the WITH MOVE option.

If you add the replica using the ALTER AVAILABILITY GROUP Transact-SQL command or using Windows PowerShell cmdlets, you must prepare a database manually, as above, before joining the database to the availability group.

For more information, see:

- Select Initial Data Synchronization Page (AlwaysOn Availability Group Wizards);
- RESTORE (Transact-SQL);
- Start Data Movement on an AlwaysOn Secondary Database;

Join a Database to an Availability Group

If a database was manually prepared by restoring a backup from the primary replica on a secondary replica, it must join the availability group as soon as possible. This starts data movement from the primary replica to the secondary.

To join the database, you must be connected to the SQL Server instance hosting the secondary replica and the secondary replica must already be added to the availability group.

For more information, see Join a Secondary Database to an Availability Group.

SQL Server System Data and Availability Groups

An availability group replica does not contain system metadata by default. Data such as login IDs, server roles, and server-level permissions are contained in each Master database of each SQL Server instance.

An administrator must prepare each secondary replica with the necessary system data manually or must make the availability group databases contained databases. Contained databases are a SQL Server 2012 feature allowing system metadata to be stored in user databases, reducing their dependency on the Master database and reducing cross-server administration.

For more information, see:

- Contained Databases;
- Contained database authentication Server Configuration Option.

If you elect not to use contained databases, you will have to transfer login IDs and passwords from the primary replica SQL Server to the secondary replica SQL Servers. For more information, see How to transfer logins and passwords between instances of SQL Server.

Jobs, Alerts, Email Operators

Transfer any jobs, schedules, alerts, and operators from the primary replica that will need to run on the secondary replicas. The secondary replicas are read-only, but you may choose to perform full database backups and transaction log backups on them as well as reporting.

There is a SQL Server Integration Services (SSIS) Transfer Jobs task just for this purpose. This will require some development work, setting up the task with a source and destination Connection Manager. For more information, see Transfer Jobs Task. You can also use Windows PowerShell to script SQL Server Agent jobs.

On the secondary replicas, the SQL Server Agent service should be configured using a Windows domain user account that is a member of the sysadmin fixed server role in the SQL Server instance. It is a best practice to use the same account for all SQL Server Agent instances in the availability group.

The SQL Server Agent service should not be configured to auto-restart SQL Server or the SQL Server Agent service on a failover cluster instance.

For more information, see:

- SQL Server Agent;
- Select an Account for the SQL Server Agent Service;
- Configure SQL Server Agent.

Backups

SQL Server Availability Groups, though they use the transaction log to transfer data between replicas, do not take the place of SQL Server backups. There should be periodic full database backups, optionally differential backups, and transaction log backups in each replica database the same as any other production database.

When the transaction log is backed up, it is truncated—and the space occupied by the backed up data is available for new transactions. If the transaction log is not backed up periodically, it will continue to grow and consume storage.

You have the option of doing backups on read-only replicas in order to minimize resource consumption on the primary replica. See Read-Only Secondary Replicas and Application Intent, below.

Backups from designated replicas can still form a complete backup and restore chain.

For more information, see Backup Overview.

Availability Group Listeners

An availability group listener is an essential part of an availability group deployment.

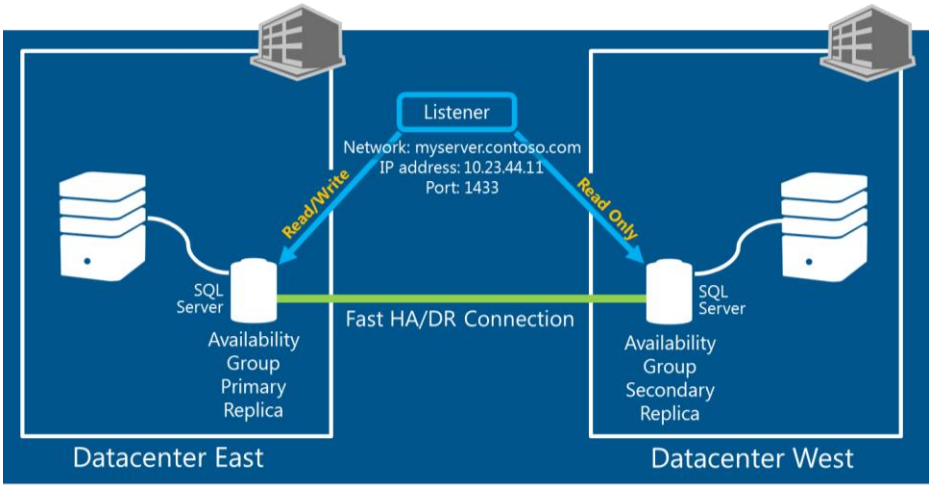

Figure 8: *The Availability Group Listener is a network name, IP address, and port that follows the primary replica from node to node after a failover. If read-only routing is configured, a .NET connection string indicating a read-only client will be automatically routed to a secondary replica.*

A listener is virtual network name that points to a primary or secondary replica of an availability group. By using the listener name, a client does not have to know the network name of the SQL Server instance to which the client is connecting. A listener will automatically follow the primary replica from node to node after a failover.

Availability group listener properties include the DNS network name, the TCP port on which the client must connect to the availability group replicas, and/or more IP addresses. IP addresses can be specified for different subnets. The network name must be unique in the domain and in NetBIOS.

The availability group listener is a resource in the cluster. It is a dependency of the availability group.

When the client attempts a connection to the listener, DNS resolves the name into all the specified IP addresses. The client will attempt to connect to one IP address after the other until a connection succeeds or the attempt times out. This happens automatically with a .NET client. The logic does not have to be coded.

If an availability group exists on a single subnet, you can configure the Listener IP addresses to use DHCP; however, this is not recommended. Using static IP addresses removes, for example, the possibility that an expired IP address will cause transactions to rollback.

Port 1433 is the default for SQL Server. If this port is designated as the Listener port, client connection strings do not have to specify a port. Listener ports other than 1433 must be specified in the client connection string.

If the connection string does not specify read-only access to the database, the listener connection will be to the primary replica.

When the primary replica fails over to a secondary replica, the client will be disconnected while the failover takes place. After the failover is complete, the client can reconnect to the new primary replica using the listener network name and port.

A client can specify the physical SQL Server instance network name and instance name even though it hosts an availability group. A client does not have to use the listener name. For example, there may be other production databases on a SQL Server instance besides the Availability Group databases. Though the client cannot write to a secondary replica, the client can write to other databases on a SQL Server instance that are not part of an availability group secondary replica.

For more information, see:

- Create or Configure an Availability Group Listener;
- Availability Group Listeners, Client Connectivity, and Application Failover.

Read-Only Replicas and Application Intent

A primary replica is the only replica where write operations can take place in the availability group databases. If this is a high volume transaction processing database, there can be blocking locks between reads and writes or between writes and other writes. In order to keep these blocks to a minimum, read operations can take place on a secondary replica.

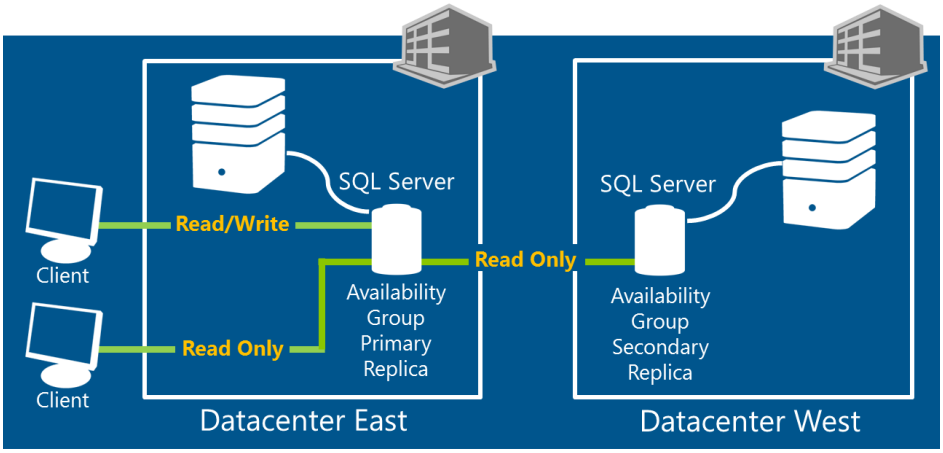

***Figure 9**: Read Only Routing allows clients that specify an ApplicationIntent=Read Only connection string to be automatically routed to a secondary replica. This allows read-only reporting, queries, etc. to be isolated from primary replica writing and reduce blocking locks.*

Backups, reporting, and ad-hoc queries can all be done on a read-only replica and will not interfere with write activity on the primary replica. In order to optimize performance of searches in the read-only replica databases, SQL Server will automatically create and update temporary index distribution statistics. These temporary statistics are stored in the Tempdb database so no changes are required in the replica database. This allows the SQL Server Query Optimizer to generate optimal query plans on the secondary replica as it would do on the primary replica while not requiring any user intervention.

By default, no connections are allowed on secondary replicas. Both read-only access and read-only routing must be configured before clients can connect to secondary replicas.

You can specify whether a secondary replica can accept read-only connections when adding a secondary replica to the Availability Group, or you can change the property after the secondary replica joins the availability group.

When you use the CREATE AVAILABILITY GROUP or ALTER AVAILABILITY GROUP Transact-SQL commands, specify an ALLOW_CONNECTIONS parameter of NO, READ_ONLY, or ALL. With a secondary replica, even if you specify all connections, they will be read-only.

Next, specify a read-only routing URL for each secondary replica SQL Server instance to be configured. An Availability Group Listener must already exist. Using CREATE AVAILABILITY GROUP or ALTER AVAILABILITY GROUP, this is the SECONDARY_ROLE option.

A read-only routing URL is the URL that client connections will use to connect to the secondary instance. It is not the same as the endpoint URL. The read-only routing URL will usually use port 1433 as the default instance of SQL Server running on that cluster node and assigned that IP address.

Next, specify the read-only routing list for the primary replica. Using CREATE AVAILABILITY GROUP or ALTER AVAILABILITY GROUP, this is the PRIMARY_ROLE option. This is a list of secondary replica SQL Server instances to which read-only client connections will be routed. The order of the list is the order in which secondary servers will be used.

After read-only routing is configured, clients can use the ApplicationIntent=ReadOnly property in a connection string along with the Listener network name and port. This will automatically route the client connection to a read-only replica.

Hotfixes for .NET 3.5 SP1 and .NET 4.0 are available for support of the ApplicationIntent property.

For more information, see:

- Configure Read-Only Access on an Availability Replica;
- Configure Read-Only Routing for an Availability Group;
- Availability Group Listeners, Client Connectivity, and Application Failover;
- AlwaysOn Solution Guide: Offloading Read-Only Workloads to Secondary Replicas
- Client Connection Access to Availability Replicas.

Multi-Subnet Failover

Starting with SQL Server 2012 SP1, a client can specify the MultiSubnetFailover property in a connection string. Setting this property value to TRUE or YES will cause failovers across subnets to be faster. The client will attempt to connect to all available IP addresses and will connect to the first one available.

Hotfixes for .NET 3.5 SP1 and .NET 4.0 are available for support of the MultiSubnetFailover property.

For more information, see:

- SQL Server Multi-Subnet Clustering;
- Availability Group Listeners, Client Connectivity, and Application Failover.

Operational Considerations

The administrator has many tools available with which to monitor Availability Groups. SQL Server Management Studio has a dashboard; Transact-SQL commands, Windows PowerShell cmdlets, and a Microsoft System Center Management Pack can all be used. System Center is the recommended method of monitoring availability groups.

There are also System Monitor objects and counters with which to measure Availability Group performance.

If availability group replicas are part of a replication topology, automatic failover of a publisher is supported. Only manual failover of a subscriber is supported. Failover of a distribution server is not supported.

A planned failover, such as for applying service packs or upgrades, is distinguished from a forced manual failover. A planned manual failover is carried out between the primary replica and a secondary replica configured in synchronous commit mode.

A forced manual failover is intended only for disaster recovery, when a planned manual failover is not possible. There may be data loss as a result of a forced manual failover. A

forced manual failover is carried out between a primary replica and a secondary replica configured in asynchronous commit mode.

For more information, see:

- Administration of an Availability Group;
- Monitoring of Availability Groups;
- Perform a Planned Manual Failover of an Availability Group;
- Perform a Forced Manual Failover of an Availability Group.

Summary

With careful planning, AlwaysOn Availability Groups can be a valuable high availability and disaster recovery option that Hosted SQL Server companies can offer their customers.