2017: Always On Failover Cluster Instances using Windows Server Failover Clustering (WSFC)

As part of the SQL Server Always On offering, Always On Failover Cluster Instances leverages Windows Server Failover Clustering (WSFC) functionality to provide local high availability through redundancy at the server-instance level—a failover cluster instance (FCI).

An FCI is a single instance of SQL Server that is installed across Windows Server Failover Clustering (WSFC) nodes and, possibly, across multiple subnets.

On the network, an FCI appears to be an instance of SQL Server running on a single computer, but the FCI provides failover from one WSFC node to another if the current node becomes unavailable.

SQL Failover Cluster Instance

An FCI consists of a set of physical servers (nodes) that contain similar hardware configuration as well as identical software configuration that includes operating system version and patch level, and SQL Server version, patch level, components, and instance name. Identical software configuration is necessary to ensure that the FCI can be fully functional as it fails over between the nodes.

WSFC Resource Group

A SQL Server FCI runs in a WSFC resource group. Each node in the resource group maintains a synchronized copy of the configuration settings and check-pointed registry keys to ensure full functionality of the FCI after a failover, and only one of the nodes in the cluster owns the resource group at a time (the active node).

The WSFC service manages the server cluster, quorum configuration, failover policy, and failover operations, as well as the VNN and virtual IP addresses for the FCI.

In case of a failure (hardware failures, operating system failures, application or service failures) or a planned upgrade, the resource group ownership is moved to another node in the FCI.

The number of nodes that are supported in a WSFC resource group depends on your SQL Server edition.

Also, the same WSFC cluster can run multiple FCIs (multiple resource groups), depending on your hardware capacity, such as CPUs, memory, and number of disks.

Storage

Contrary to the availability group, an FCI must use shared storage between all nodes of the FCI for database and log storage.

The shared storage can be in the form of WSFC cluster disks, disks on a SAN, Storage Spaces Direct (S2D), or file shares on an SMB.

This way, all nodes in the FCI have the same view of instance data whenever a failover occurs. This does mean, however, that the shared storage has the potential of being the single point of failure, and FCI depends on the underlying storage solution to ensure data protection.

To install or upgrade a SQL Server failover cluster, you must run the Setup program on each node of the failover cluster. To add a node to an existing SQL Server failover cluster, you must run SQL Server Setup on the node that is to be added to the SQL Server failover cluster instance. Do not run Setup on the active node to manage the other nodes.

Considerations to the "Farm Owner":

1. Commit Mode:
    a. High-availability vs.: (Asynchronous commit: when the availability replicas are distributed over considerable distances.)
        i. Does not wait for any secondary replica to write incoming transaction log records to disk (to harden the log).
        ii. The primary replica does not wait for that secondary replica to harden the log.
    b. High-safety: (synchronous-commit mode, transactions wait to send the transaction confirmation to the client until the secondary replica has hardened the log to disk at the cost of latency in performance.)
        i. Both the primary replica and a given secondary replica are both configured for synchronous-commit mode,
        ii. The primary replica waits for the secondary replica to confirm that it has hardened the log.
2. Apply Commit Mode to replicas to create best solution for co-located or distributed server farms.
    a. Primary replica
    b. Secondary replica.
3. Transaction log behavior equals disk space
4. Indexes:
    a.
5. Other

Applies To: SQL Server 2016 Preview

The AlwaysOn Availability Groups feature is a high-availability and disaster-recovery solution that provides an enterprise-level alternative to database mirroring. Introduced in SQL Server 2012, AlwaysOn Availability Groups maximizes the availability of a set of user databases for an enterprise.

**An availability group supports a failover environment for a discrete set of user databases, known as availability databases that fail over together.

An availability group supports a set of read-write primary databases and one to eight sets of corresponding secondary databases. Optionally, secondary databases can be made available for read-only access and/or some backup operations.

An availability group fails over at the level of an availability replica. Failovers are not caused by database issues such as a database becoming suspect due to a loss of a data file, deletion of a database, or corruption of a transaction log.

In AlwaysOn Availability Groups, the availability mode is a replica property that determines whether a given availability replica can run in synchronous-commit mode.

For each availability replica, the availability mode must be configured for either synchronous-commit mode or asynchronous-commit mode.

If the primary replica is configured for asynchronous-commit mode, it does not wait for any secondary replica to write incoming transaction log records to disk (to harden the log).

If a given secondary replica is configured for asynchronous-commit mode, the primary replica does not wait for that secondary replica to harden the log.

If both the primary replica and a given secondary replica are both configured for synchronous-commit mode, the primary replica waits for the secondary replica to confirm that it has hardened the log (unless the secondary replica fails to ping the primary replica within the primary's session-timeout period).

The key components are as follows:

- Supports up to nine availability replicas. An *availability replica* is an instantiation of an availability group that is hosted by a specific instance of SQL Server and maintains a local copy of each availability database that belongs to the availability group. Each availability group supports one primary replica and up to eight secondary replicas. For more information, seeOverview of AlwaysOn Availability Groups (SQL Server).

Important
Each availability replica must reside on a different node of a single Windows Server Failover

Clustering (WSFC) cluster. For more information about prerequisites, restrictions, and recommendations for availability groups, see Prerequisites, Restrictions, and Recommendations for AlwaysOn Availability Groups (SQL Server).

Supports alternative availability modes, as follows:

- *Asynchronous-commit mode*. This availability mode is a disaster-recovery solution that works well when the **availability replicas are distributed over considerable distances**.
  - If every secondary replica is running under asynchronous-commit mode, the primary replica does not wait for any of the secondary replicas to harden the log. Rather, immediately after writing the log record to the local log file, the primary replica sends the transaction confirmation to the client.
  - The **primary replica** runs with minimum transaction latency in relation to a secondary replica that is configured for asynchronous-commit mode.
  - If the **current primary** is configured for asynchronous commit availability mode, it will commit transactions asynchronously for all secondary replicas regardless of their individual availability mode settings.
- *Synchronous-commit mode*. This availability mode emphasizes high availability and data protection over performance, at the cost of increased transaction latency.
  - A given availability group can support up to three synchronous-commit availability replicas, including the current primary replica.
  - Under synchronous-commit mode, transactions wait to send the transaction confirmation to the client until the secondary replica has hardened the log to disk. When data synchronization begins on a secondary database, the secondary replica begins applying incoming log records from the corresponding primary database.
  - As soon as every log record has been hardened, the secondary database enters the SYNCHRONIZED state. Thereafter, every new transaction is hardened by the secondary replica before the log record is written to the local log file.
  - When all the secondary databases of a given secondary replica are synchronized, synchronous-commit mode supports manual failover and, optionally, automatic failover.

  For more information, see Availability Modes (AlwaysOn Availability Groups).
- Supports several forms of availability-group failover: automatic failover, planned manual failover (generally referred as simply "manual failover"), and forced manual failover (generally referred as simply "forced failover"). For more information, see Failover and Failover Modes (AlwaysOn Availability Groups).
- Enables you to configure a given availability replica to support either or both of the following active-secondary capabilities:
  - Read-only connection access which enables read-only connections to the replica to access and read its databases when it is running as a secondary replica. For more information, see Active Secondaries: Readable Secondary Replicas (AlwaysOn Availability Groups).
  - Performing backup operations on its databases when it is running as a secondary replica. For more information, seeActive Secondaries: Backup on Secondary Replicas (AlwaysOn Availability Groups).

Using active secondary capabilities improves your IT efficiency and reduce cost through better resource utilization of secondary hardware. In addition, offloading read-intent applications and backup jobs to secondary replicas helps to improve performance on the primary replica.

- Supports an availability group listener for each availability group. An *availability group listener* is a server name to which clients can connect in order to access a database in a primary or secondary replica of an AlwaysOn availability group. Availability group listeners direct incoming connections to the primary replica or to a read-only secondary replica. The listener provides fast application failover after an availability group fails over. For more information, see Availability Group Listeners, Client Connectivity, and Application Failover (SQL Server).
- Supports a flexible failover policy for greater control over availability-group failover. For more information, see Failover and Failover Modes (AlwaysOn Availability Groups).
- Supports automatic page repair for protection against page corruption. For more information, see Automatic Page Repair (Availability Groups/Database Mirroring).
- Supports encryption and compression, which provide a secure, high performing transport.
- Provides an integrated set of tools to simplify deployment and management of availability groups, including:
    - Transact-SQL DDL statements for creating and managing availability groups. For more information, see Overview of Transact-SQL Statements for AlwaysOn Availability Groups (SQL Server).
    - SQL Server Management Studio tools, as follows:
        - The New Availability Group Wizard creates and configures an availability group. In some environments, this wizard can also automatically prepare the secondary databases and start data synchronization for each of them. For more information, see Use the New Availability Group Dialog Box (SQL Server Management Studio).
        - The Add Database to Availability Group Wizard adds one or more primary databases to an existing availability group. In some environments, this wizard can also automatically prepare the secondary databases and start data synchronization for each of them. For more information, see Use the Add Database to Availability Group Wizard (SQL Server).
        - The Add Replica to Availability Group Wizard adds one or more secondary replicas to an existing availability group. In some environments, this wizard can also automatically prepare the secondary databases and start data synchronization for each of them. For more information, see Use the Add Replica to Availability Group Wizard (SQL Server Management Studio).
        - The Fail Over Availability Group Wizard initiates a manual failover on an availability group. Depending on the configuration and state of the secondary replica that you specify as the failover target, the wizard can perform either a planned or forced manual failover. For more information, see Use the Fail Over Availability Group Wizard (SQL Server Management Studio).
    - The AlwaysOn Dashboard monitors AlwaysOn availability groups, availability replicas, and availability databases and evaluates results for AlwaysOn policies. For more information, see Use the AlwaysOn Dashboard (SQL Server Management Studio).
    - The Object Explorer Details pane displays basic information about existing availability groups. For more information, see Use the Object Explorer Details to Monitor Availability Groups (SQL Server Management Studio).

- PowerShell cmdlets. For more information, see Overview of PowerShell Cmdlets for AlwaysOn Availability Groups (SQL Server).

Terms and Definitions:

Availability group

A container for a set of databases, *availability databases*, that fail over together.

Availability database

A database that belongs to an availability group. For each availability database, the availability group maintains a single read-write copy (the *primary database*) and one to eight read-only copies (*secondary databases*).

Primary database

The read-write copy of an availability database.

Secondary database

A read-only copy of an availability database.

Availability replica

An instantiation of an availability group that is hosted by a specific instance of SQL Server and maintains a local copy of each availability database that belongs to the availability group. Two types of availability replicas exist: a single *primary replica* and one to eight *secondary replicas*.

Primary replica

The availability replica that makes the primary databases available for read-write connections from clients and, also, sends transaction log records for each primary database to every secondary replica.

Secondary replica

An availability replica that maintains a secondary copy of each availability database, and serves as a potential failover targets for the availability group. Optionally, a secondary replica can support read-only access to secondary databases can support creating backups on secondary databases.

The AlwaysOn Availability Groups active secondary capabilities include support for read-only access to one or more secondary replicas (readable secondary replicas). A readable secondary replica allows read-only access to all its secondary databases. However, readable secondary databases are not set to read-only.

They are dynamic. A given secondary database changes as changes on the corresponding primary database are applied to the secondary database. For a typical secondary replica, the data, including durable memory optimized tables, in the secondary databases is in near real time.

Furthermore, full-text indexes are synchronized with the secondary databases. In many circumstances, data latency between a primary database and the corresponding secondary database is only a few seconds.

Security settings that occur in the primary databases are persisted to the secondary databases. This includes users, database roles, and applications roles together with their respective permissions and transparent data encryption (TDE), if enabled on the primary database.

Indexing

To optimize read-only workloads on the readable secondary replicas, you may want to create indexes on the tables in the secondary databases. Because you cannot make schema or data changes on the secondary databases, create indexes in the primary databases and allow the changes to transfer to the secondary database through the redo process.

To monitor index usage activity on a secondary replica, query the user_seeks, user_scans, and user_lookups columns of the sys.dm_db_index_usage_stats dynamic management view.

Statistics for Read-Only Access Databases

Statistics on columns of tables and indexed views are used to optimize query plans. For availability groups, statistics that are created and maintained on the primary databases are automatically persisted on the secondary databases as part of applying the transaction log records. However, the read-only workload on the secondary databases may need different statistics than those that are created on the primary databases. However, because secondary databases are restricted to read-only access, statistics cannot be created on the secondary databases.

To address this problem, the secondary replica creates and maintains temporary statistics for secondary databases in tempdb. The suffix _readonly_database_statistic is appended to the name of temporary statistics to differentiate them from the permanent statistics that are persisted from the primary database.

Only SQL Server can create and update temporary statistics. However, you can delete temporary statistics and monitor their properties using the same tools that you use for permanent statistics:

Delete temporary statistics using the DROP STATISTICS Transact-SQL statement.

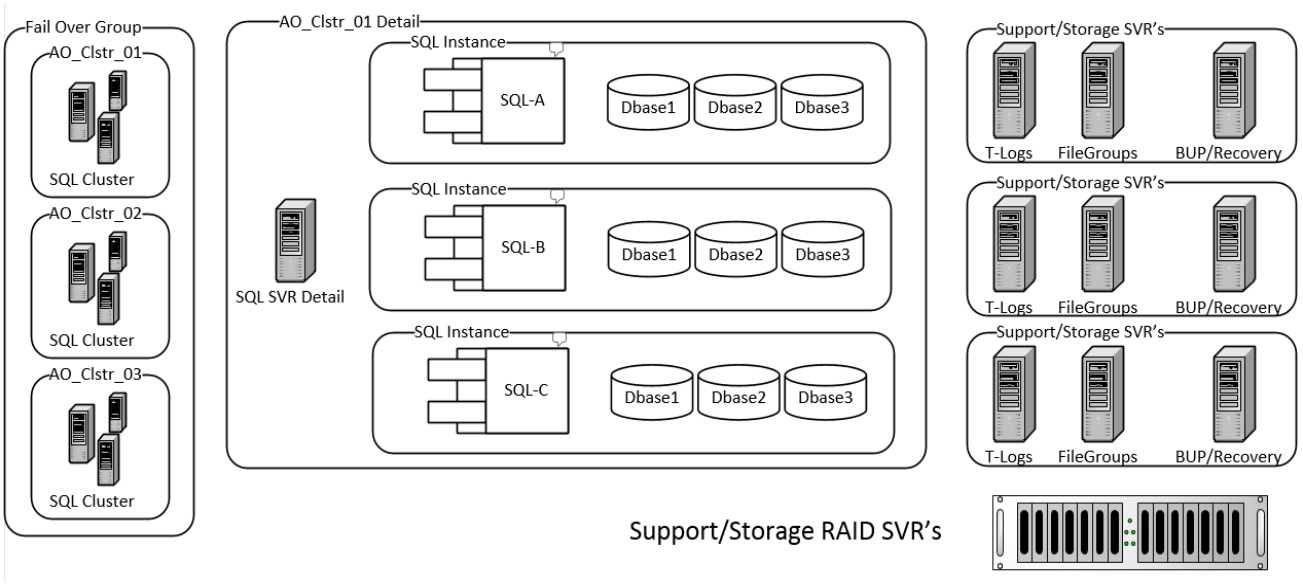
Monitor statistics using the sys.stats and sys.stats_columns catalog views. sys_stats includes a column, is_temporary, to indicate which statistics are permanent and which are temporary.

There is no support for auto-statistics update for memory-optimized tables on the primary or secondary replica. You must monitor query performance and plans on the secondary replica and manually update the statistics on the primary replica when needed. However, the missing statistics are automatically created both on primary and secondary replica.

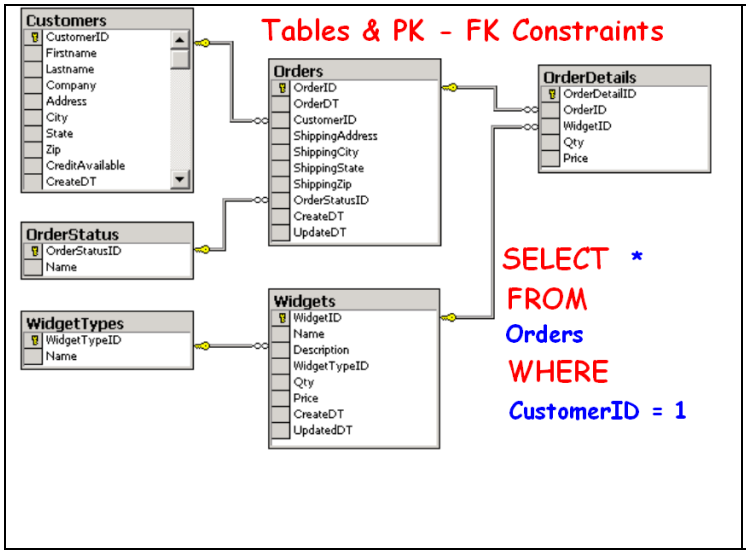
Availability group listener

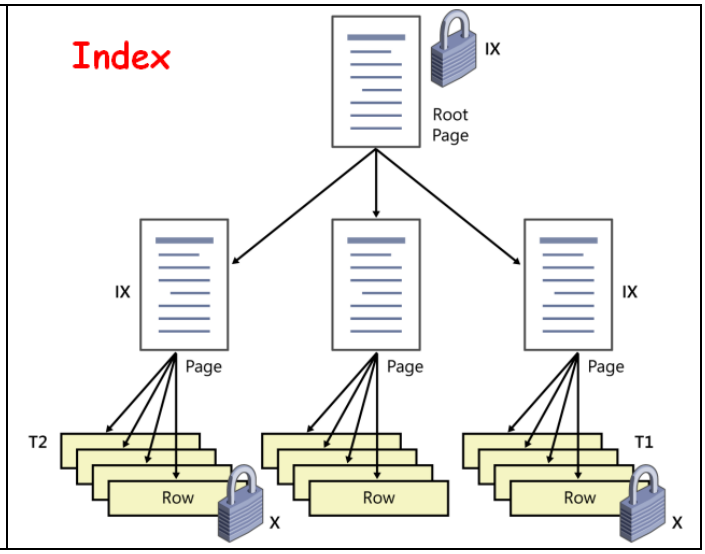
A server name to which clients can connect in order to access a database in a primary or secondary replica of an AlwaysOn availability group. Availability group listeners direct incoming connections to the primary replica or to a read-only secondary replica.

1. AlwaysOn Availability Groups (SQL Server): https://msdn.microsoft.com/library/hh510230.aspx
   a. https://msdn.microsoft.com/en-us/library/ff877931.aspx
2. Always On Failover Group – contains 3 Failover Clusters – each cluster server may have several "Instances" – each instance may have several Support/Storage RAID arrays for database filegroups, T-log shipping, scheduled weekly backup/recovery jobs…
3. Each instance is a "Service" requiring a domain account to work. Each instance has its own SQL Agent for jobs…



Each **Database** is comprised of **Tables** & **Rows** – each defined by rows with a defined **Data Type**, possible **Primary Key** or **Foreign Key** "Constraint", **Relation** to other tables and **NULL** handling (is the value allowed to be "Empty"?)	An "**Index**" (like in a book) help SQL Server find data without having to read every value in each row/table. Clustered indexes are created when a Primary Key is defined and stored in that order. Other indexes can also be created.





AlwaysOn Failover Cluster Instances

As part of the SQL Server AlwaysOn offering, AlwaysOn Failover Cluster Instances leverages Windows Server Failover Clustering (WSFC) functionality to provide local high availability through redundancy at the server-instance level—a failover cluster instance (FCI). An FCI is a single instance of SQL Server that is installed across Windows Server Failover Clustering (WSFC) nodes and, possibly, across multiple subnets. On the network, an FCI appears to be an instance of SQL Server running on a single computer, but the FCI provides failover from one WSFC node to another if the current node becomes unavailable.

AlwaysOn Availability Groups

AlwaysOn Availability Groups is an enterprise-level high-availability and disaster recovery solution introduced in SQL Server 2012 to enable you to maximize availability for one or more user databases. AlwaysOn Availability Groups requires that the SQL Server instances reside on Windows Server Failover Clustering (WSFC) nodes.

AlwaysOn Availability Groups, the high availability and disaster recovery solution introduced in SQL Server 2016, requires Windows Server Failover Clustering (WSFC). Also, though AlwaysOn Availability Groups is not dependent upon SQL Server Failover Clustering, you can use a failover clustering instance (FCI) to host an availability replica for an availability group. It is important to know the role of each clustering technology, and to know what considerations are necessary as you design your AlwaysOn Availability Groups environment.

Deploying AlwaysOn Availability Groups requires a Windows Server Failover Clustering (WSFC) cluster. To be enabled for AlwaysOn Availability Groups, an instance of SQL Server must reside on a WSFC node, and the WSFC cluster and node must be online. Furthermore, each availability replica of a given availability group must reside on a different node of the same WSFC cluster. The only exception is that while being migrated to another WSFC cluster, an availability group can temporarily straddle two clusters.

AlwaysOn Availability Groups relies on the Windows Failover Clustering (WSFC) cluster to monitor and manage the current roles of the availability replicas that belong to a given availability group and to determine how a failover event affects the availability replicas. A WSFC resource group is created for every availability group that you create. The WSFC cluster monitors this resource group to evaluate the health of the primary replica.

The quorum for AlwaysOn Availability Groups is based on all nodes in the WSFC cluster regardless of whether a given cluster node hosts any availability replicas. In contrast to database mirroring, there is no witness role in AlwaysOn Availability Groups.

The overall health of a WSFC cluster is determined by the votes of quorum of nodes in the cluster. If the WSFC cluster goes offline because of an unplanned disaster, or due to a persistent hardware or communications failure, manual administrative intervention is required.

A Windows Server or WSFC cluster administrator will need to force a quorum and then bring the surviving cluster nodes back online in a non-fault-tolerant configuration.

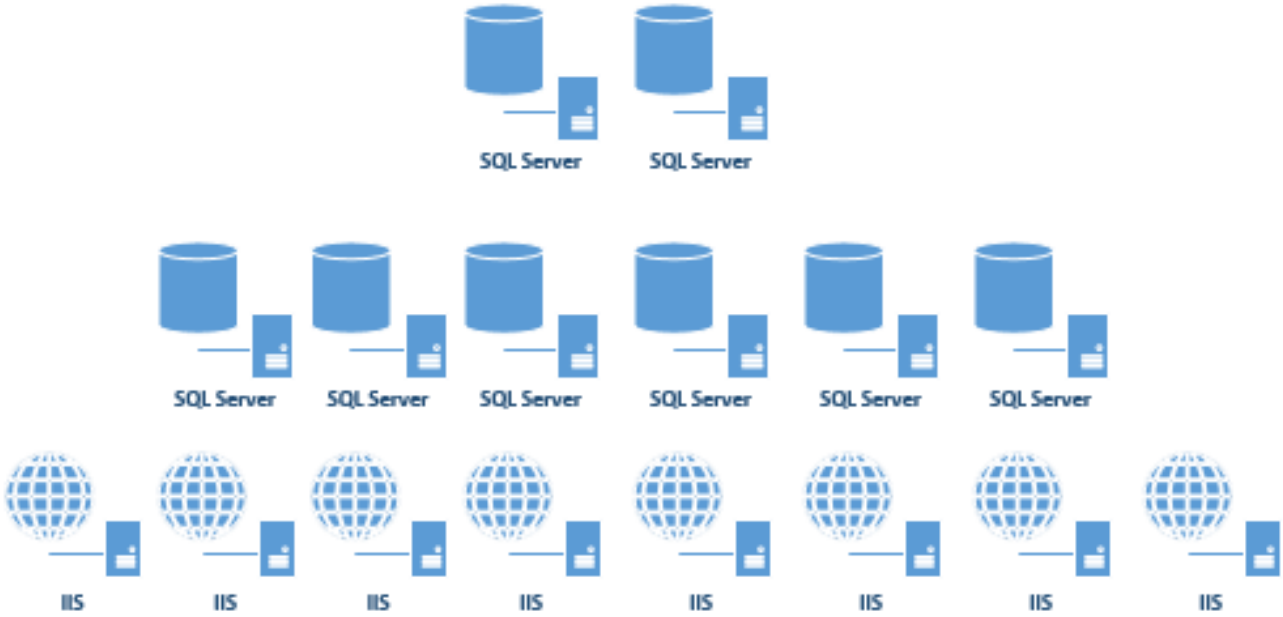
AlwaysOn Failover Cluster Instances (SQL Server)

As part of the SQL Server AlwaysOn offering, AlwaysOn Failover Cluster Instances leverages Windows Server Failover Clustering (WSFC) functionality to provide local high availability through redundancy at the server-instance level—a failover cluster instance (FCI).

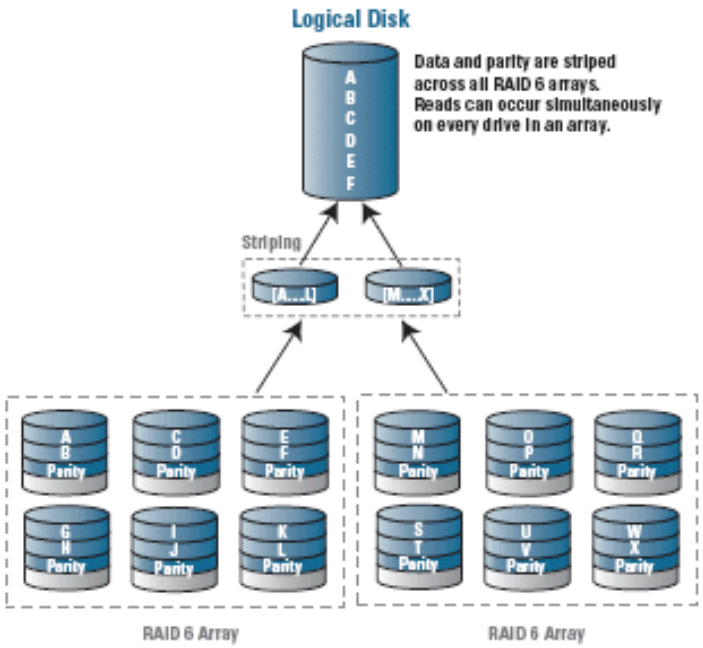
An FCI is a single instance of SQL Server that is installed across Windows Server Failover Clustering (WSFC) nodes and, possibly, across multiple subnets. On the network, an FCI appears to be an instance of SQL Server running on a single computer, but the FCI provides failover from one WSFC node to another if the current node becomes unavailable. https://msdn.microsoft.com/en-us/library/ms189134.aspx

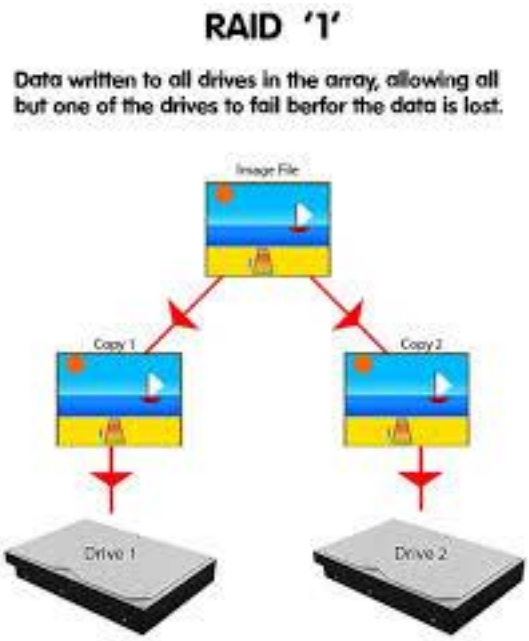
Beginning in SQL Server 2014, AlwaysOn Failover Cluster Instances supports Clustered Shared Volumes (CSV) in both Windows Server 2008 R2 and Windows Server 2012.

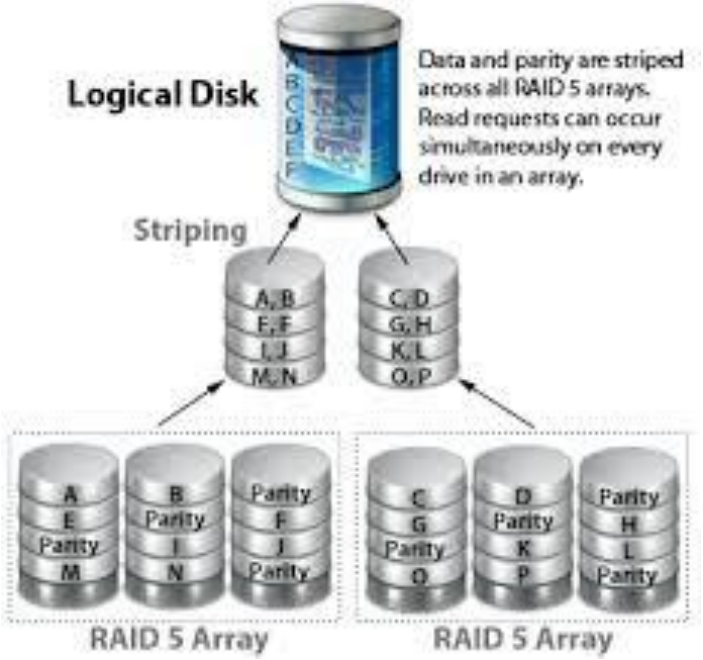
My Farm


SQL Server
SQL Server
SQL Server
SQL Server
SQL Server
SQL Server
SQL Server
SQL Server
IIS
IIS
IIS
IIS
IIS
IIS
IIS
IIS

Server Internals – each server has a hardware design



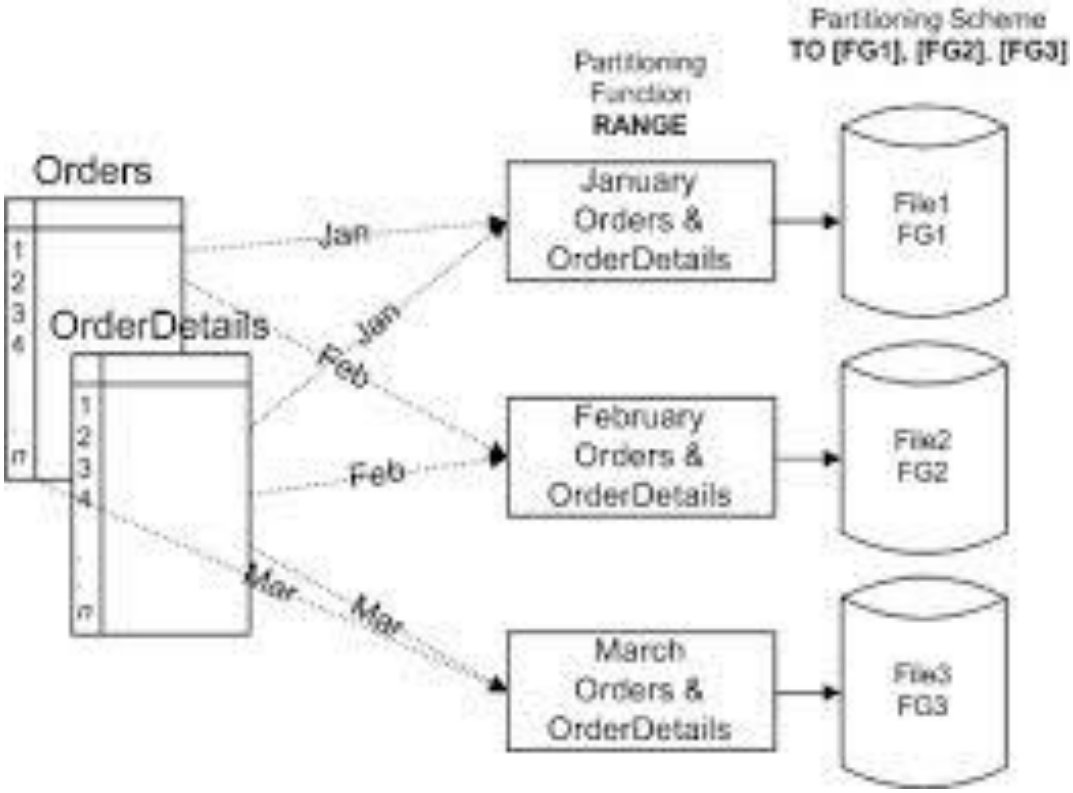




Server Replicate/Mirror

HA/AO Clustering & Grouping

Model over schedule PIT(Point in Time) window:



Subnet discussion

You have a new project that requires a primary database server in one building and secondary database server in another building to be configured as a Failover Cluster. With this setup the participating nodes will have different subnets. You propose to use the SQL Server 2012 AlwaysOn feature and you are well aware that multiple subnets are supported by the AlwaysOn feature. You may have already setup AlwaysOn in SQL Server 2012, but what needs to be done across multiple subnets? What are the important points to consider for this scenario?

Solution

One of the enhancements to AlwaysOn FCI is the support for multi subnet cluster setup. The ability to implement a multi-subnet FCI has always been included in older versions of SQL Server, but it required a virtual local area network (VLAN). SQL Server 2012 removes that requirement, enabling multi-subnet FCI to be more commonly adopted as a high availability and disaster recovery solution.

In this tip, I share several points to consider when implementing this solution.

SQL Edition

You will need SQL Server 2012 Enterprise Edition for a multi subnet cluster setup.

Storage

Multi subnet FCI will require storage level replication to maintain a copy of the database at the disaster recovery site. Though you have separate copies of the database file at each site, to SQL Server this looks like a shared storage solution.

When choosing storage replication technology, you must ensure that it meets the requirements for SQL Server storage and that the database and logs are synchronously and consistently replicated. The key concern here is that it must adhere to the write ordering requirements of SQL Server and to make sure that the data is consistent even in the event of network failure.

File Share Witness

The purpose of the FSW is to have something else that can count as a vote in situations where the number of configured nodes isn't quite enough for determining a quorum. A FSW is more likely to be used in multi-site clusters or where there is no common storage. A FSW does not store cluster configuration data like a disk. It does, however, contain information about which version of the cluster configuration database is most recent. Other than that, the FSW is just a share.

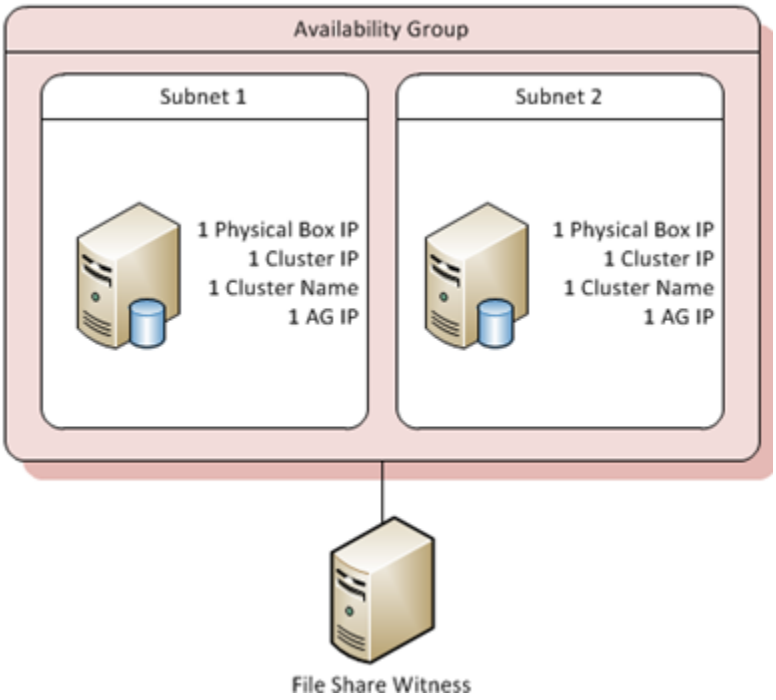
When choosing that FSW location, it should be outside of the cluster, not on either node participating in the cluster. The location must also not be found in the SAN. This actually happened with me during my setup. I realized that the path created by our SAN team could not resolve when I continued with my AlwaysOn configuration, so I asked for a shared path folder instead.

IP Addresses

SQL Server 2012 now supports failover cluster nodes residing on different subnets. The introduction of "OR" logic allows the use of 2 IP addresses. This means that IP addresses can reside in different subnets, practically eliminating the need for a VLAN.

From the diagram below, on each side of your subnet, you will need to request the following from your Domain or Network Administrator:

- Cluster IP for each node
- Cluster name for each node
- Availability Group IP for each node
- Availability Group name



Now, in order for SQL Server 2012 to deliver very high availability, these IP addresses will be registered with DNS by default.

Although you are using multiple subnets, the nodes participating in your cluster must still reside in the same domain.

Network

If you are doing the setup of AlwaysOn in a multi site cluster, make sure to tune the heartbeat network traffic.

The heartbeat frequency, also called subnet delay, is once every second regardless of the subnet configuration. By default, if a node misses a series of 5 heartbeats, another node initiates a failover. The range for this value, called subnet threshold, is 3 to 10.

Determine the value for the Time-to-Live (TTL) setting in the DNS. By default this value is set to 20 minutes (1200 seconds). For SQL Server 2012, we recommend that this value be changed to 60 seconds. You may run the cluster command below to do this:

cluster /cluster: ClusterName res "NetworkNameResource" /priv HostRecordTTL=60

If you are not familiar with your Network Name Resource, like me, you can use this command:

cluster res

It should display all the network resource available to you. Look for the SQL Network Name value and that should be it.

Use this command to check the new value of your new TTL setting:

cluster /cluster:<ClusterName> res "NetworkNameResource" /priv

Another setting to update is the DNS replication delay. This value is set to 180 minutes by default. You will also need to set the replication frequency to at least 15 minutes.

In the event of fail over to another subnet, the online IP address changes accordingly. The Windows Server failover cluster issues a DNS update as soon as the network name resource comes online. But these changes do not take effect immediately. This is because of the local DNS cache on the client machines and also because of the DNS replication latency if the cluster node and the client machines are not using the same DNS server. Therefore, the client machines cannot resolve the network name, and until the DNS replication is complete and the local DNS cache times out causing connection failures.

References

1. Always On Failover Cluster Instances (SQL Server): https://docs.microsoft.com/en-us/sql/sql-server/failover-clusters/windows/always-on-failover-cluster-instances-sql-server
2. Create a New SQL Server Failover Cluster (Setup) https://docs.microsoft.com/en-us/sql/sql-server/failover-clusters/install/create-a-new-sql-server-failover-cluster-setup
3. Failover Clustering and Always On Availability Groups (SQL Server): https://docs.microsoft.com/en-us/sql/database-engine/availability-groups/windows/failover-clustering-and-always-on-availability-groups-sql-server
4. Active Secondaries: Readable Secondary Replicas (AlwaysOn Availability Groups): https://msdn.microsoft.com/en-us/library/ff878253.aspx