**What is a cluster?**

A server cluster is a group of independent servers running Windows Server 2003, Enterprise Edition and working together as a single system to provide high availability of services for clients

When a failure occurs on one computer in a cluster, resources are redirected and the workload is redistributed to another computer in the cluster

**What is AlwaysOn Availability Group?**

AlwaysOn technology is the process of having one primary replica server that can have its primary database(s) mirrored up to four separate secondary replicas in different locations

Unlike database mirroring that has its limitations, AlwaysOn features combines the failover clustering and database mirroring technologies and goes one step further by providing read only database on the secondary replicas!!!

**Some of the limitation that database mirroring had are resolved by AlwaysOn:**

* Only one single failover partner at a time (1 to 1)
* Doesn’t protect complex multiple-database applications such as SharePoint, which had many databases that need to failover simultaneously
* Had to choose between synchronously or asynchronously options but not at the same time
* The databases on the mirror server are in a state of constant recovery, thus cannot access the data, leaving a server inactive

**With AlwaysOn, the advantages are as follows:**

* Can failover multiple databases as a unit for complex applications
* Can use either synchronously or asynchronously options on either replica
* Allows automatic failover of groups of related or non-related databases
* The secondary replicas can unload of the primary replica with backups, DBCC commands, and reporting

**Prerequisites:**

Must install Window Failover cluster Services on each node

Must have a shared folder for each server to access for backups

Must take a full backup of the primary database(s)

All server must be in full recovery mode

SET UP CLUSTER

**Availability Databases features**

* When adding the **database** to the group it must be an **online**
* **Data synchronization** is the process by which any change to the primary database is reproduced on the secondary database
* The **role** of a given replica determines whether it hosts **read-write databases or read-only databases**
* The primary replica hosts read-write database
* The secondary replica hosts read-only databases
  + Under **asynchronous**-commit mode, the primary replica commits transactions without waiting for acknowledgement that an asynchronous-commit secondary replica has hardened the log. Some data loss possible.
  + Under **synchronous**-commit mode, before committing transactions, a synchronous-commit primary replica waits for a synchronous-commit secondary replica to acknowledge that it has finished hardening the log. committed transactions are fully protected
* During a failover the primary and the secondary servers change roles
* Three forms of failover exist—automatic, manual, and forced (with possible data loss)
* You can **provide client connectivity to the primary replica** of a given availability group by **creating an availability group listener**.
* An availability group listener provides a set of resources that is attached to a given availability group to **direct client connections to the appropriate availability replica.**
* **An availability group listener is associated with a unique DNS name that serves as a virtual network name (VNN), one or more virtual IP addresses (VIPs), and a TCP port number**

--REGISTER ALL SERVERS FIRST

--NOTE: DATABASE CREATED WITH FULL RECOVERY MODE

CREATE DATABASE [Rep1]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'Rep1',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep1.mdf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

LOG ON

( NAME = N'Rep1\_log',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep1\_log.ldf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

GO

USE [master]

GO

ALTER DATABASE [Rep1] SET RECOVERY FULL WITH NO\_WAIT

GO

--NOTE: DATABASE CREATED WITH SIMPLE RECOVERY MODE

CREATE DATABASE [Rep2]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'Rep2',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep2.mdf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

LOG ON

( NAME = N'Rep1\_log',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep2\_log.ldf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

GO

USE [master]

GO

ALTER DATABASE [Rep2] SET RECOVERY SIMPLE WITH NO\_WAIT

GO

--NOTE: DATABASE DOES NOT HAVE A FULL BACKUP

CREATE DATABASE [Rep3]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'Rep3',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep3.mdf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

LOG ON

( NAME = N'Rep1\_log',

FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\Rep3\_log.ldf' ,

SIZE = 8192KB ,

FILEGROWTH = 65536KB )

GO

USE [master]

GO

ALTER DATABASE [Rep3] SET RECOVERY FULL WITH NO\_WAIT

GO

--TAKE FULL BACKUPS OF REP1 AND REP2 SERVER BUT NOT REP3 SERVER TO A SHARED FOLDER!!!

BACKUP DATABASE [Rep1]

TO DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\Rep1.bak'

WITH NOFORMAT, INIT

GO

BACKUP DATABASE [Rep2]

TO DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Backup\Rep2.bak'

WITH NOFORMAT, INIT

GO

--POPULATE DATA FROM ADVENTUREWORKS DATABASE TO REP1,REP2,REP3 USING EXPORT WIAZRD

--START THE ALWAYS ON SET UP

USE [master]

GO

DROP AVAILABILITY GROUP [123];

GO

restore database rep1

restore database rep4

drop database rep1

drop database rep4