



Planning and Implementing a High Availability and Disaster Recovery Environment

Introduction to high availability and disaster recovery for Azure SQL solutions, and database backup and recovery

Objectives

- The difference between recovery time and recovery point objectives
- The available HADR options
- The considerations for planning and configuring HADR solutions including how backup and restore fits in
- The factors that comprise a HADR strategy

Describe High Availability and Disaster Recovery Strategies



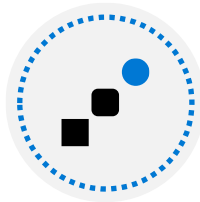
Objectives



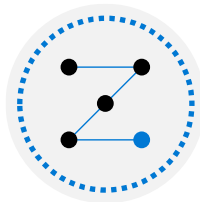
Recovery time objective (RTO)



Recovery point objective (RPO)



The available HADR options



How to devise a HADR strategy

Recovery Time Objective

Recovery Time Objective (RTO) is defined as the maximum amount of time to get a database or system up and running after an outage or problem



May have financial or business consequences if RTO is exceeded



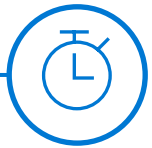
When calculating RTO you must account for processes such as copying backup files, network speed



RTO can be defined for the entire system or for individual components

Recovery Point Objective

Recovery Point Objective (RPO) represents the maximum amount of data loss that is acceptable to the business



RPO is the point in time to which a database needs to be recovered



Zero data loss may be an option but often not a guarantee

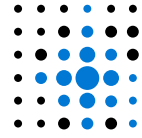


Other environmental factors interfere with realizing ideal RPO

Defining Recovery Time Objective and Recovery Point Objective



Must be defined per component (i.e. SQL Server) as well as the entire application architecture



The ability to recover from an outage is only as good as its weakest link – i.e. single points of failure



Getting everything online may take hours - RTOs and RPOs are usually separately



RTOs and RPOs should be defined for both HA and DR



DR generally involves a new data center, with different components - SQL Server is just one of them



HA is considered a more localized event

Infrastructure-as-a-Service vs. Platform-as-a-Service

IaaS solutions

You use a combination of **platform availability options**, in conjunction with **database level solutions**

In **IaaS**, there is **no built-in** high availability or disaster recovery at the database or the operating system level

PaaS solutions

In **PaaS** solutions like Azure SQL Database, and Azure SQL Managed Instance, **high availability is built into the platform**

In **PaaS and IaaS**, Disaster Recovery is an **optional** configuration

SQL Server HADR features for Azure virtual machines

- **Instance level protection** means that users and jobs are available on failover
- **Database level protection** means that everything inside of the user database is synchronized, but system objects in Master and MSDB require manual synchronization

Feature	Protects
Always On Failover Cluster Instance (FCI)	Instance
Always On Availability Group (AG)	Database
Log Shipping	Database

Both FCIs and AGs require an underlying cluster mechanism

For SQL Server deployments running on **Windows Server** use Windows Server Failover Cluster (WSFC), and for **Linux** use Pacemaker

Always On Failover Cluster Instance

Instance level protection

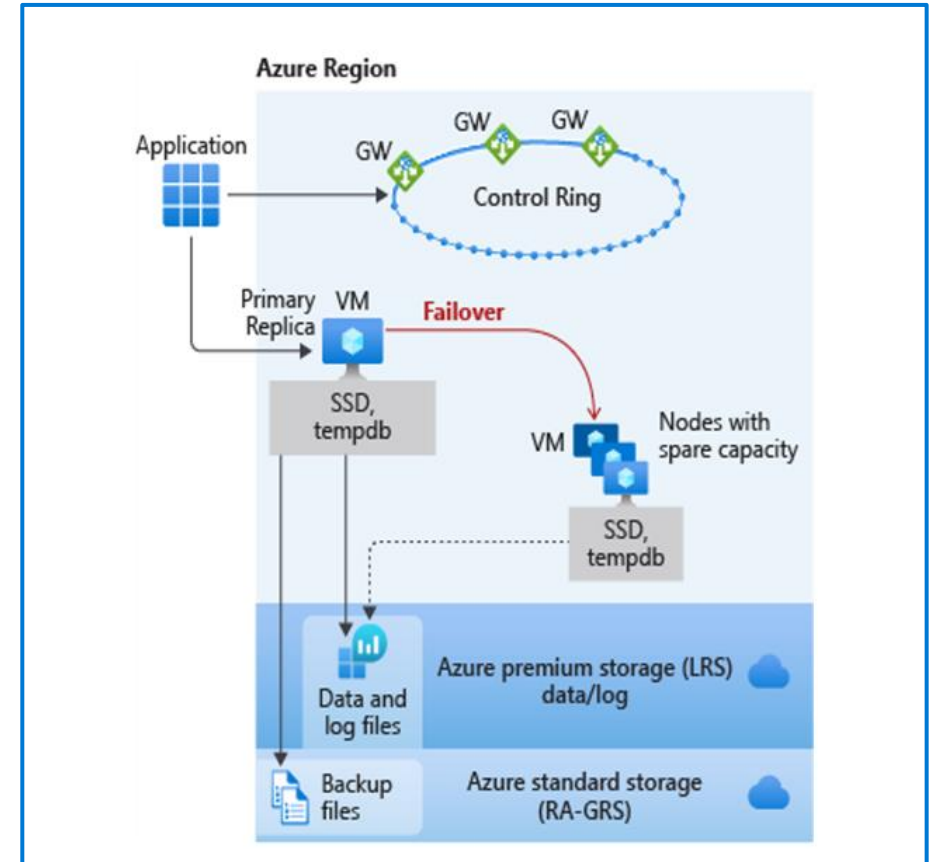
- Objects created at the instance level are protected

Failover process is a full stop and start of SQL Server

- Failover can be automatic or manual

Requires

- An underlying cluster
- A shared disk storage
- An Azure Load Balancer



Always On Availability Group

- Provides high availability, and requires an underlying cluster
- There is a read-scale option, but that model does not support failover
- Availability groups have **no shared storage requirement**, each replica maintains a full copy of each database in the group

Standard Edition

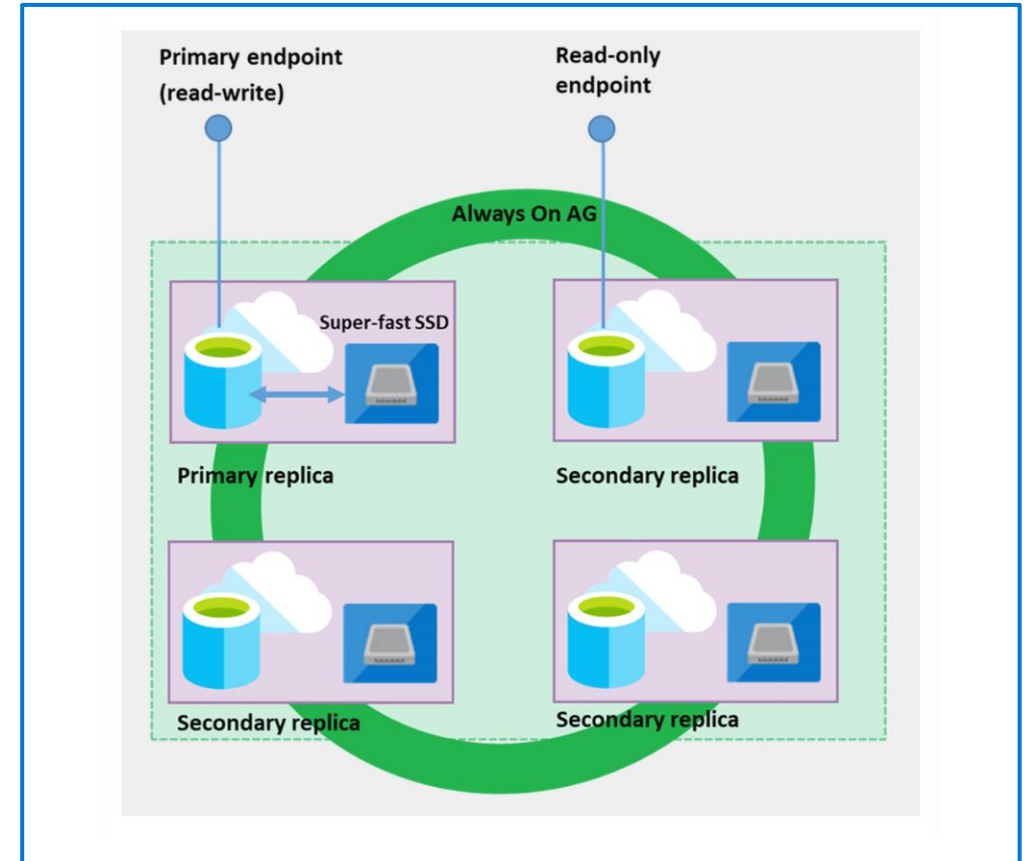
Supports only two replicas (one primary and one secondary), one database per AG

Enterprise Edition

Supports up to nine replicas per AG (one primary and up to 8 secondaries), and can have multiple databases in an AG

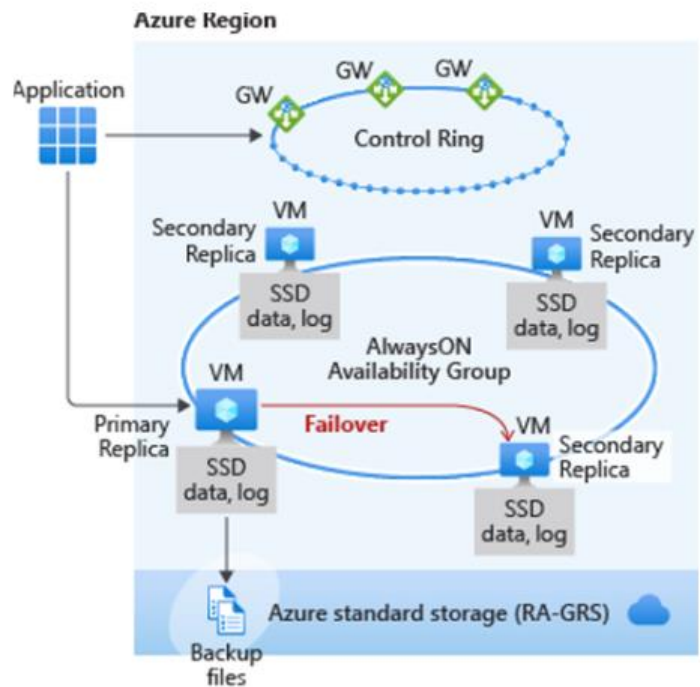
Always On Availability Group

- Can be used in **IaaS** or **PaaS**
- Database level protection
- The primary replica is read/write and the secondary replica is Read Only
- Replication can be **synchronous** or **asynchronous**.
- Use Azure Active Directory Authentication or a contained database

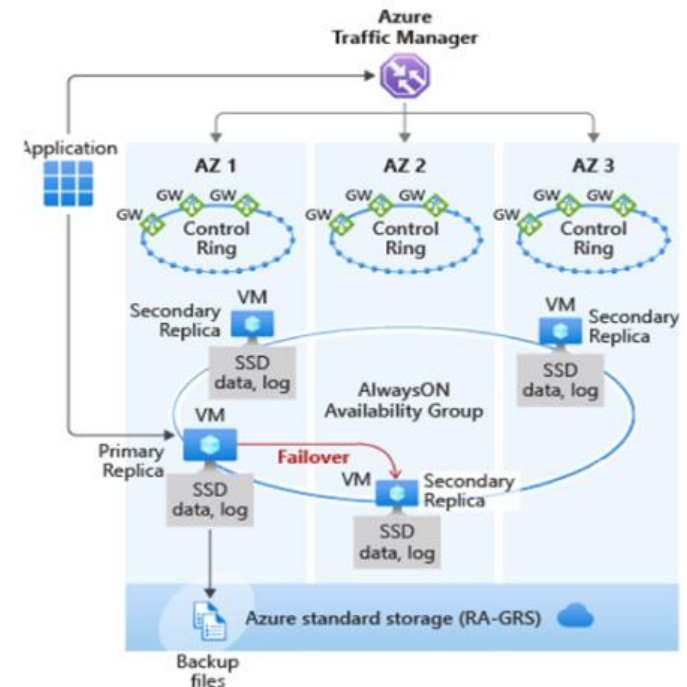


Always On Availability Group

One region



Multiple zones



Log shipping



Based on backup and restore to provide user database level protection.
Doesn't include SQL Server Agent jobs, linked servers, instance-level logins, etc.



Automated process for backing up, copying, and applying transaction log backups to provide the **simplest method** of achieving **HADR** for SQL Server.
Primarily used for DR, but it could also be used for **HA**.



Transaction log loads always slightly delayed.



Protects against human error (accidental deletion of data).

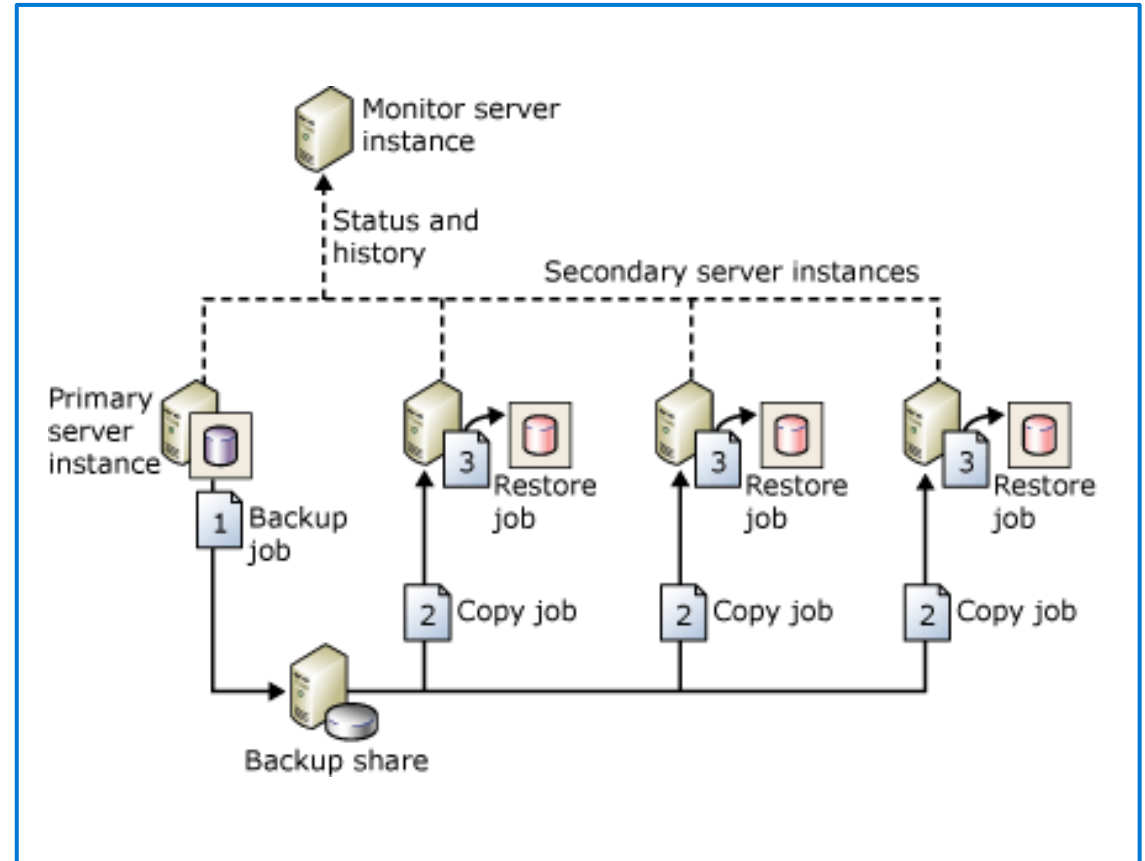
Log shipping



When switching to secondary, application must be able to tolerate a name change or use a DNS alias



Can configure automatic replication on a **schedule** to **backup, copy and restore** the primary database's transaction log to secondary



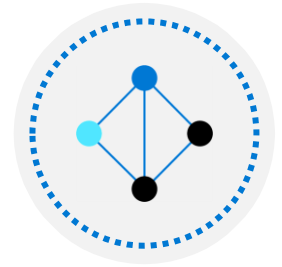
Azure HADR features for Azure Virtual Machines



Availability sets

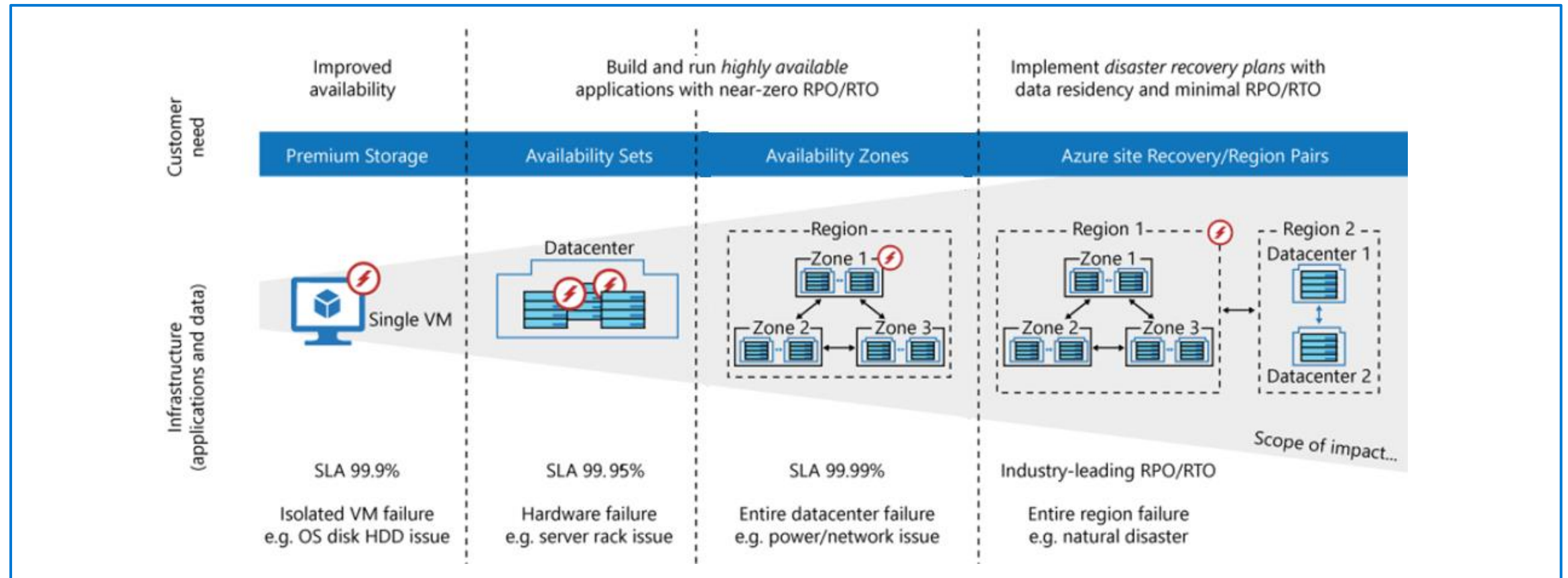


Availability zones



Azure site recovery

High Availability offered by a single VM, Availability Sets, and Availability Zones, and Azure Site Recovery



Availability sets

Protects against physical hardware failure, network outages, power interruptions and patching update outages

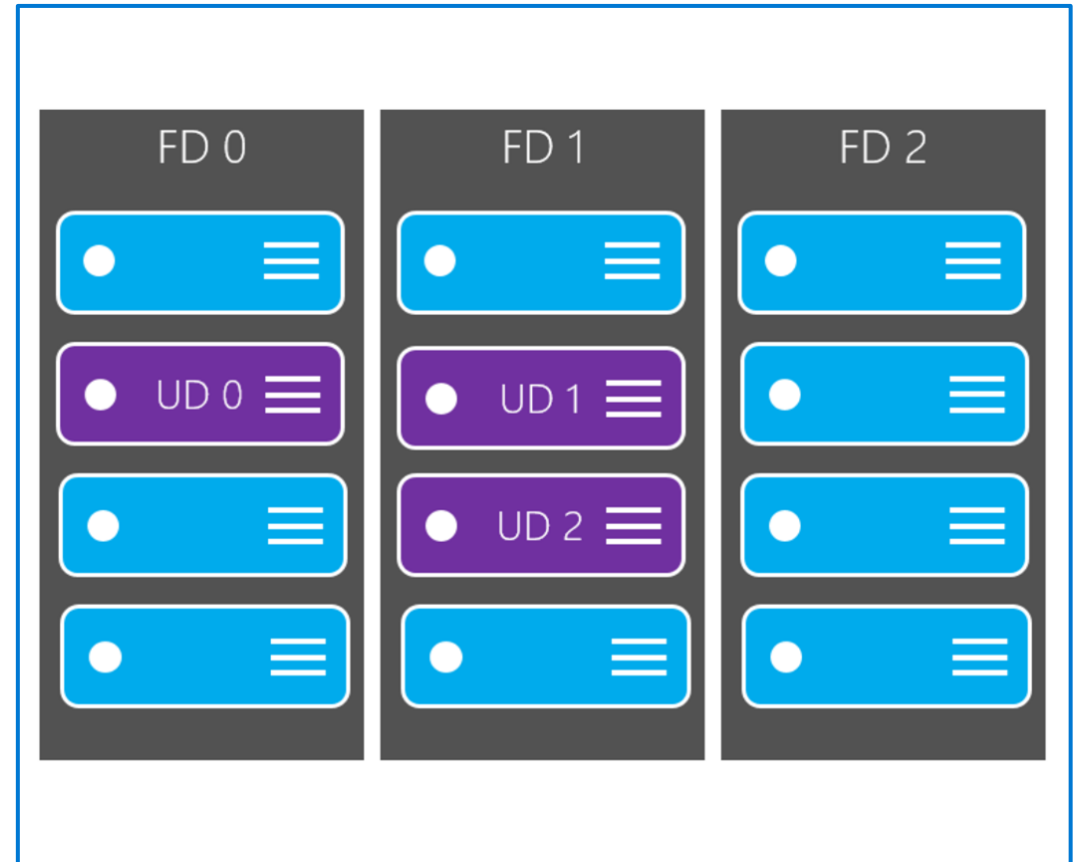
Fault domain:

- Up to 3
- VM will not share the same power source, network switch

Update domain:

- Up to 5
- Controls reboot behavior

Does not protect against a failure in the VM's OS or RDBMS



Availability Zones



VMs (or other Azure resources) are placed in separate datacenters in the same Azure region



Enables survival of a datacenter



Does not protect against a failure in the guest



Latency should be low enough to support synchronous data transfer between zones

Azure Site Recovery



SQL Server agnostic – knows nothing of SQL Server transactions



Supports hybrid disaster recovery



Replicates at the VM disk level; VMs might need to be grouped together



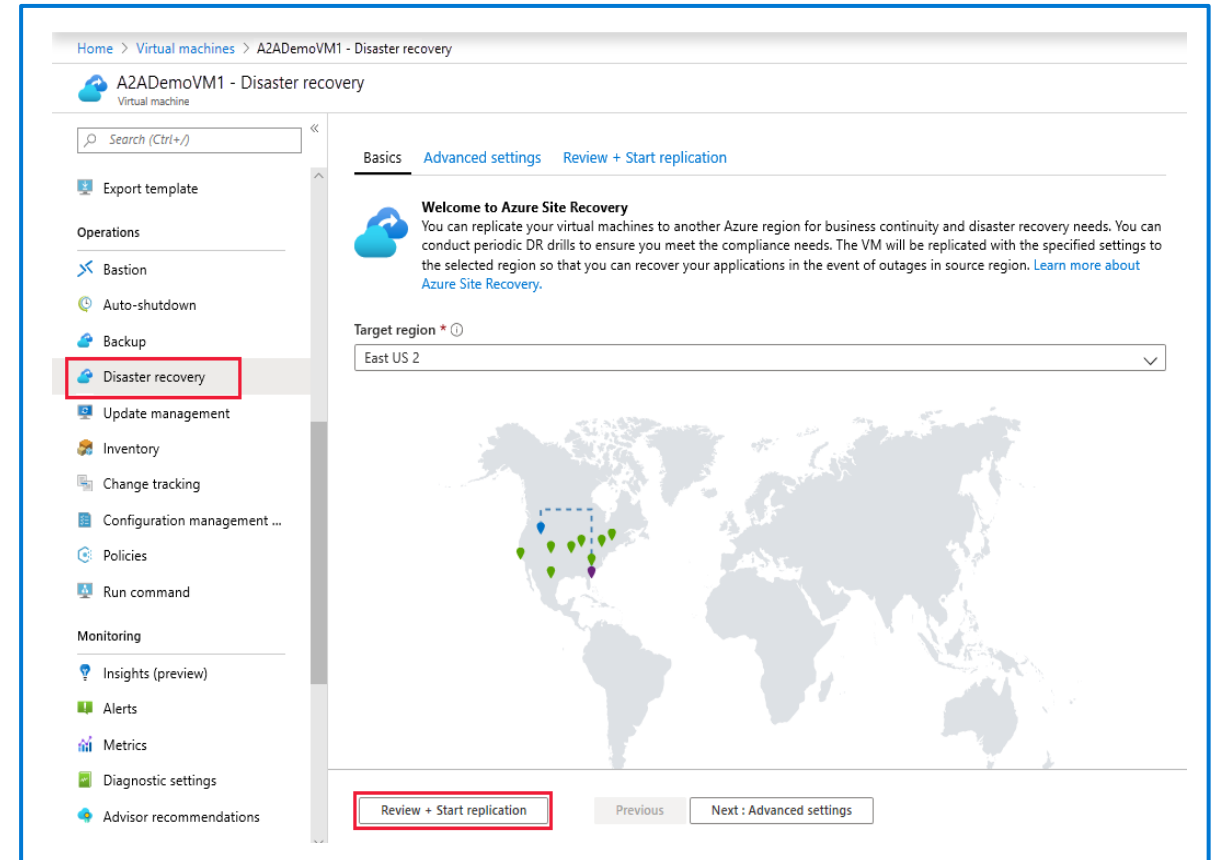
Frequently used for migration from on-premises to Azure



Can test DR without bringing down production

Azure Site Recovery – DR example

- Not a SQL Server-based disaster solution
- Most prefer a **database-centric** approach to lower RPO
- ASR can be used with more than just SQL Server
- ASR may meet RTO and RPO
- ASR is provided as part of the Azure platform

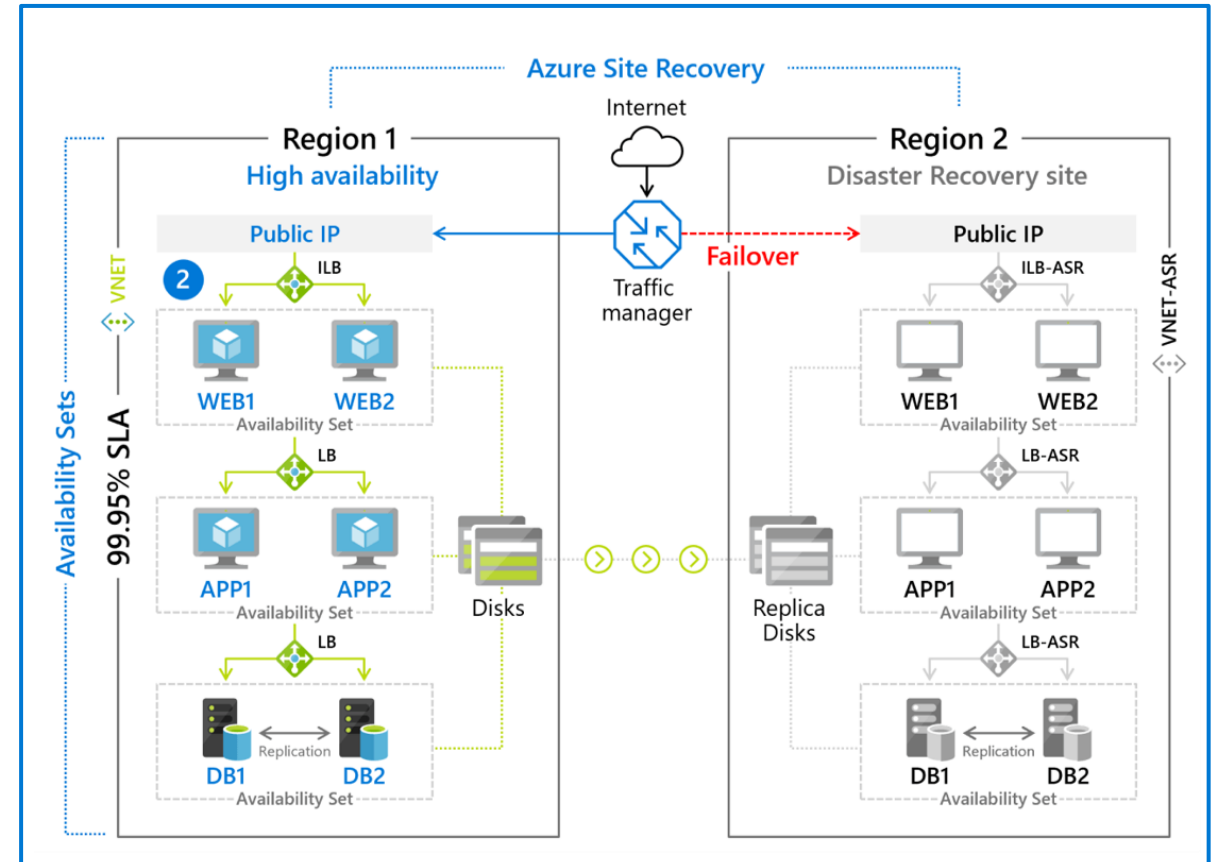


Azure Site Recovery

Replicate Azure VMs from one Azure region to another

Replicate on-premises VMware VMs, Hyper-V VMs, physical servers (Windows and Linux), Azure Stack VMs to Azure

Replicate AWS Windows instances to Azure



Azure Site Recovery cont'd

1

VM is registered with Azure Site Recovery

2

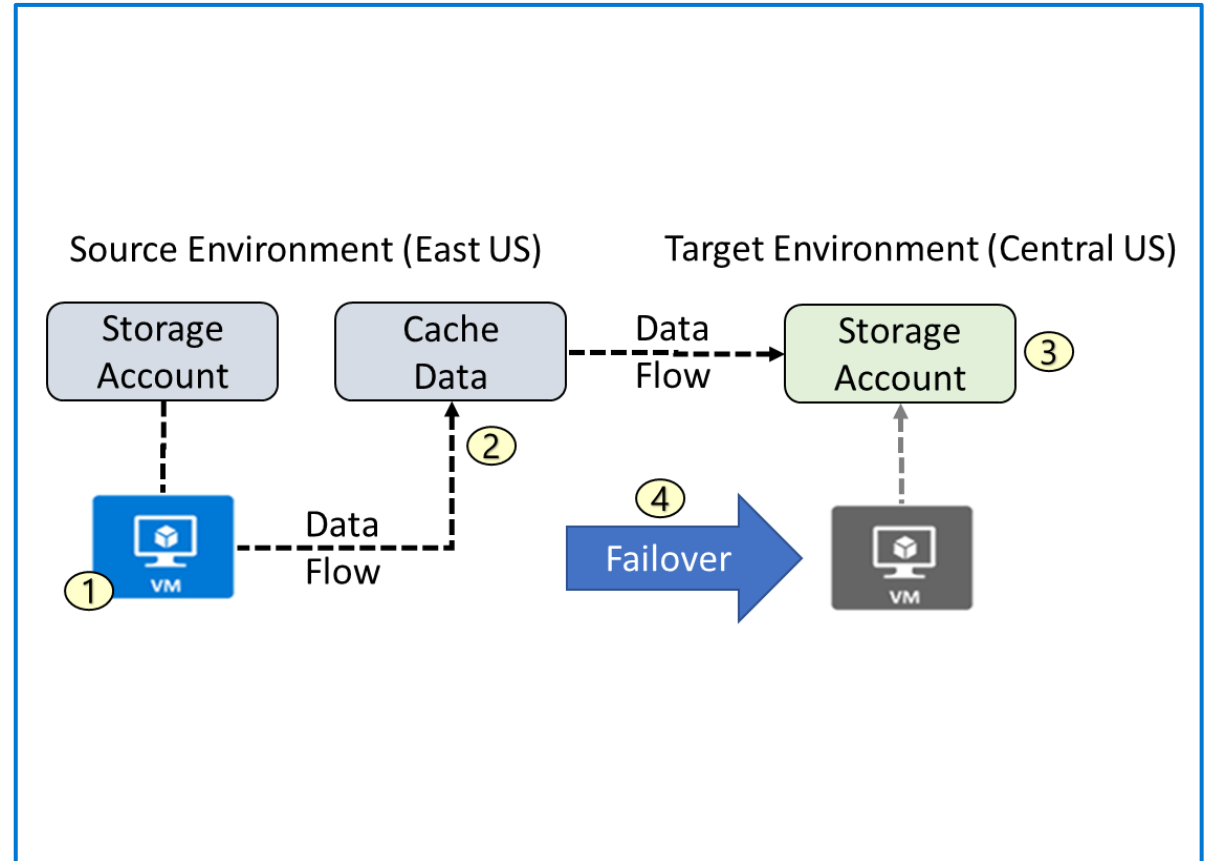
Data is continuously replicated to cache

3

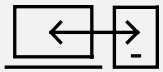
Cache is replicated to the target storage account

4

During failover the virtual machine is added to the target environment



Hybrid solutions



On premises (physical or virtual) with an Azure component:

Includes backing up to Azure as cold storage



Can go Azure to or from another cloud provider or on-premises

Networking is the most important component to consider:

- ▶ ExpressRoute from on premises to Azure
- ▶ Site to Site VPN for everything else
- ▶ Need adequate bandwidth
- ▶ Must be secure

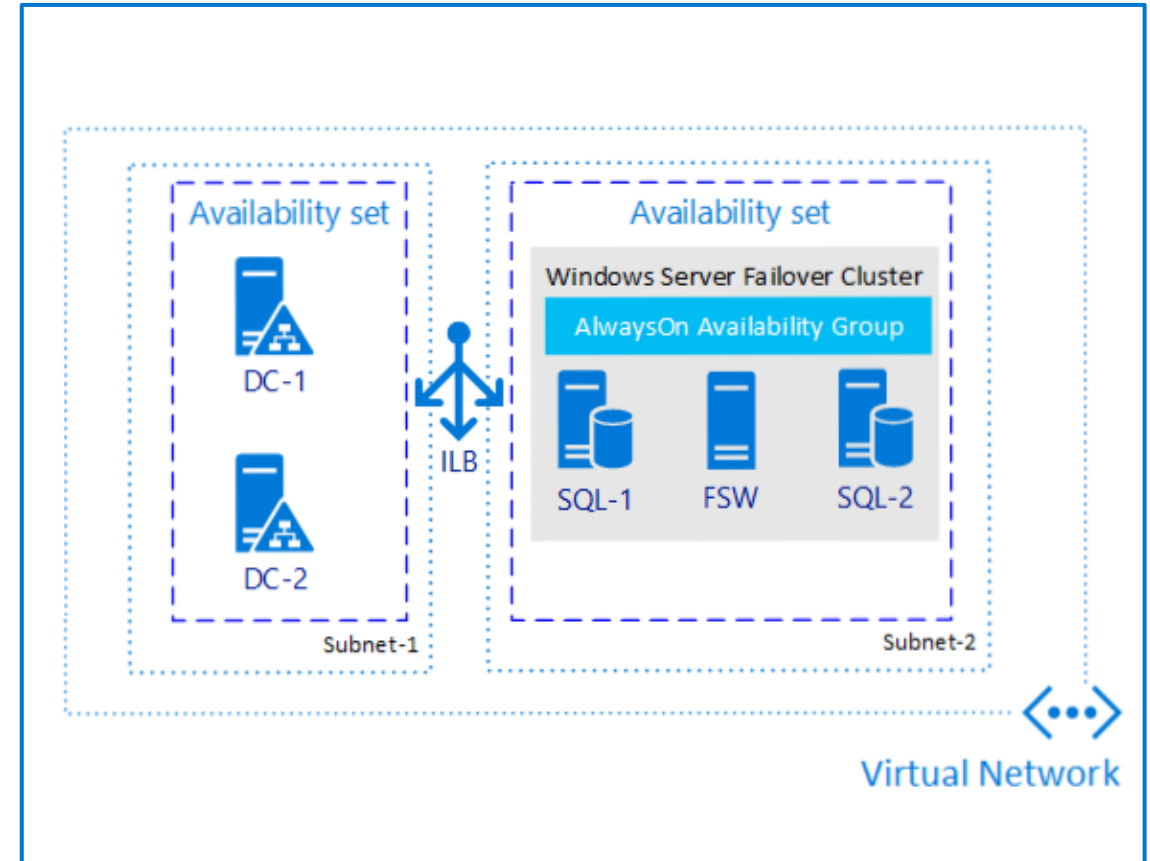
HA Single Region Example – Availability Group

Protects data by having more than one copy on different VMs

No shared storage, so easier to set up than an FCI

Provides enhanced availability during patching scenarios

Meet RTO and RPO with minimal-to-no data loss



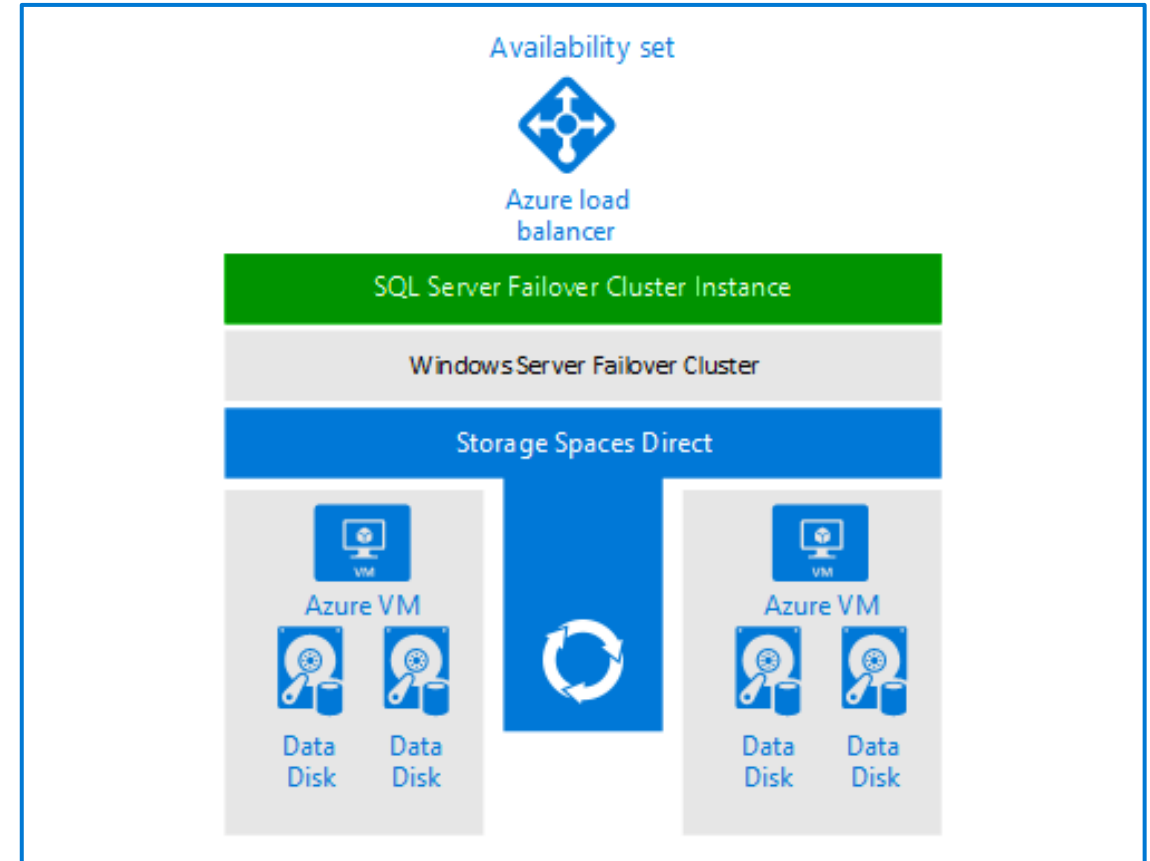
HA Single Region Example – Failover Cluster Instance

Well established, proven solution

Meets most RTO and RPO

Provides enhanced availability during patching scenarios

Designed to protect against network card failure or disk failure

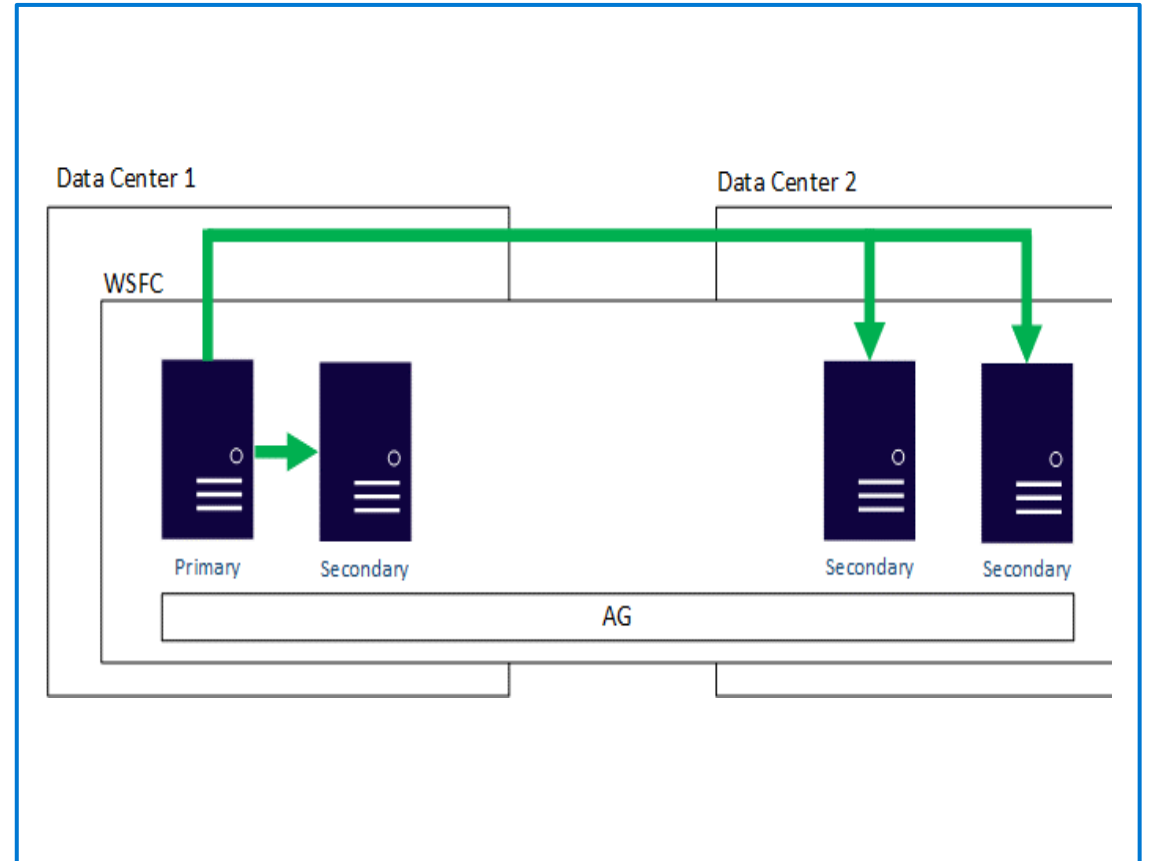


Availability Groups vs. Failover Cluster Instance

	Nodes within an FCI	Replicas within an availability group
Uses WSFC	Yes	Yes
Protection level	Instance	Database
Storage type	Shared	Non-shared
Storage solutions	Direct attached, SAN, mount points, SMB	Depends on node type
Readable secondaries	No*	Yes
Applicable failover policy settings	WSFC quorum FCI-specific Availability group settings**	WSFC quorum Availability group settings
Failed-over resources	Server, instance, and database	Database only

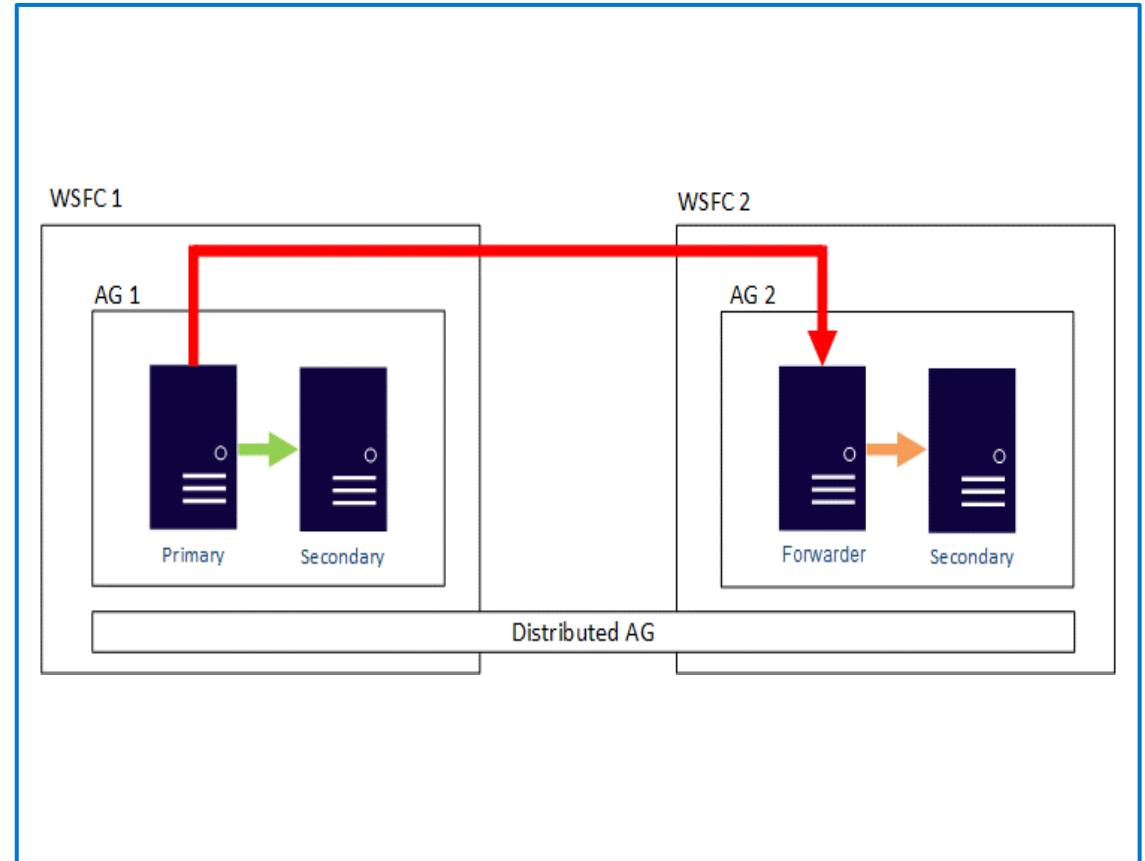
DR example – Multi-region/Hybrid Availability Group


- Use with **Standard** and **Enterprise** editions of SQL Server
- Provides both HA and DR
- **All nodes** which contain the replicas participate in the same WSFC (assumes good network connectivity)
- Consider using a cloud witness for WSFC
- Requires AD DS and DNS



DR example – Distributed Availability Group

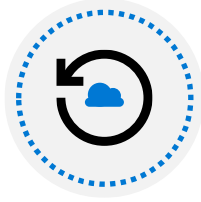
- Uses multiple WSFCs and multiple AGs
- Two WSFCs, no single point of failure
- Primary replica contains the read write database (global primary) and is **not** synchronizing all secondary replicas
- Used in **Azure** or **hybrid**, provides fail back from one location to another



The background is a dark blue gradient. On the right side, there is a glowing line that starts as a white dot, curves downwards, and then transitions into a vibrant pink and purple color. The line has a soft, ethereal glow. In the background, there are faint, embossed geometric shapes, including squares and rectangles, which add depth to the design.

Explore IaaS and PaaS Solutions for High Availability and Disaster Recovery

Objectives



What to consider when deploying a WSFC in Azure



What to consider when deploying an AG in Azure



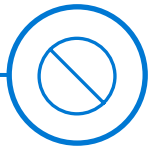
Understand active-geo replication



Explore auto-failover groups

Windows Server Failover Cluster

For AG and FCI, a cluster is required to protect against server or component failure (e.g. network failure, or RDBMS crashing)



For availability only, not for scale out performance



Enabled via the failover clustering feature



WSFCs managed via Failover Cluster Manager or PowerShell

Considerations for deploying a WSFC in Azure



Witness:

A core component of quorum which helps ensure WSFC continues to run



Unless special requirements, one vNIC for each node in Azure

WSFC name and IP Address(es):

- ▶ WSFC requires a **unique name** different than any FCI, listener, or server in the environment
- ▶ One IP address per subnet for the WSFC itself
- ▶ WSFC name will require an **AD DS object** named the Cluster Name Object (CNO)
- ▶ WSFC name and IP address get an entry in DNS

Create a Failover Cluster: Checklist

- Verify the prerequisites
- Install the Failover Clustering feature on every server that you want to add as a cluster node
- Run the Cluster Validation Wizard to validate the configuration
- Run the Create Cluster Wizard to create the failover cluster
- Create clustered roles to host cluster workloads

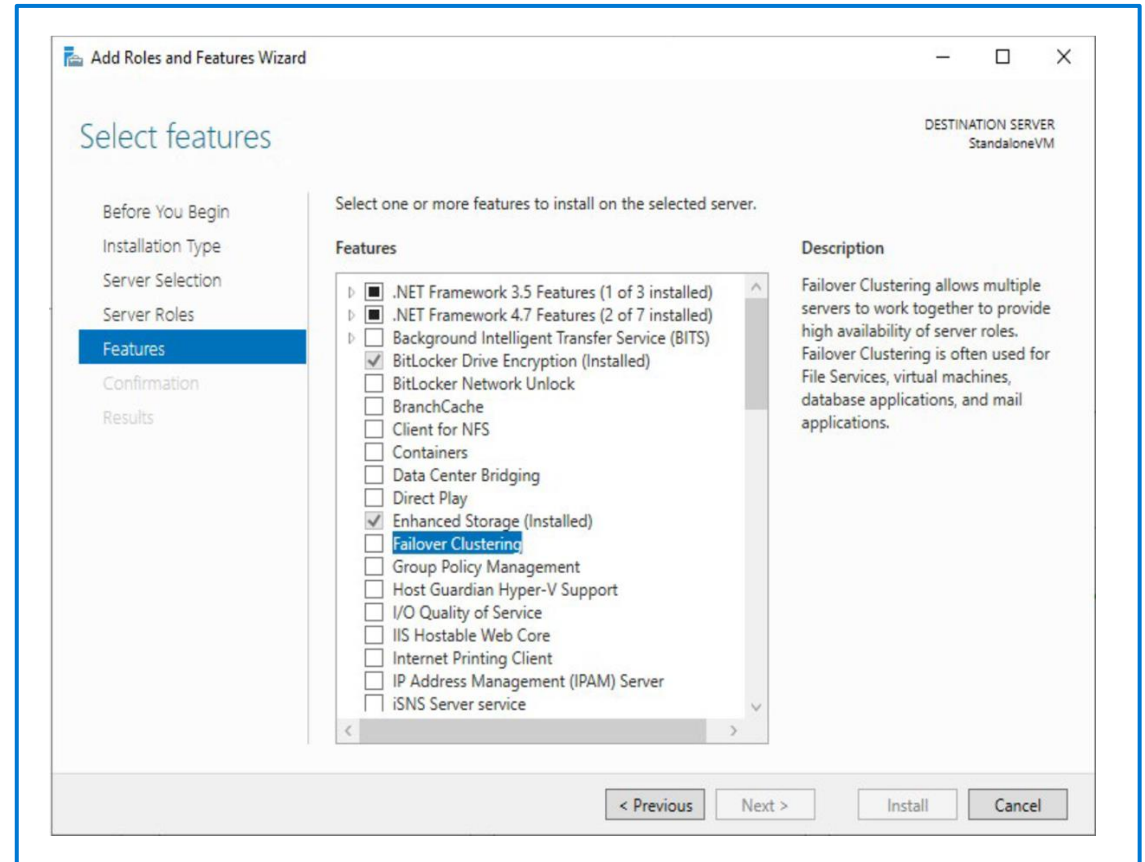
Failover Clustering Feature

Before a WSFC can be configured, the Failover Clustering feature must be enabled on every node that will use WSFC

In Server Manager, Failover Clustering must be enabled in the Add Roles and Features Wizard

In PowerShell:

```
Install-WindowsFeature Failover-Clustering -IncludeManagementTools
```



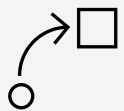
Cluster validation



Built-in process to test the configuration to ensure it is good



Required to create a WSFC



Must be run before clustering the VMs

Configuration must pass:

- ▶ **All green**
- ▶ **Warnings** only if not actually a problem or something unexpected
- ▶ **Failures** must be fixed
- ▶ **Not applicable** can be ignored

Create a Windows Server Failover Cluster

Do not use the Wizard in Failover Cluster Manager unless IP address is DHCP



Use PowerShell for static IP addresses



PowerShell Example
with shared storage:

```
New-Cluster -Name MyWSFC -Node Node1,Node2,...,NodeN  
-StaticAddress w.x.y.z
```



PowerShell Example
with no shared storage:

```
New-Cluster -Name MyWSFC -Node Node1,Node2,...,NodeN  
-StaticAddress w.x.y.z -NoStorage
```



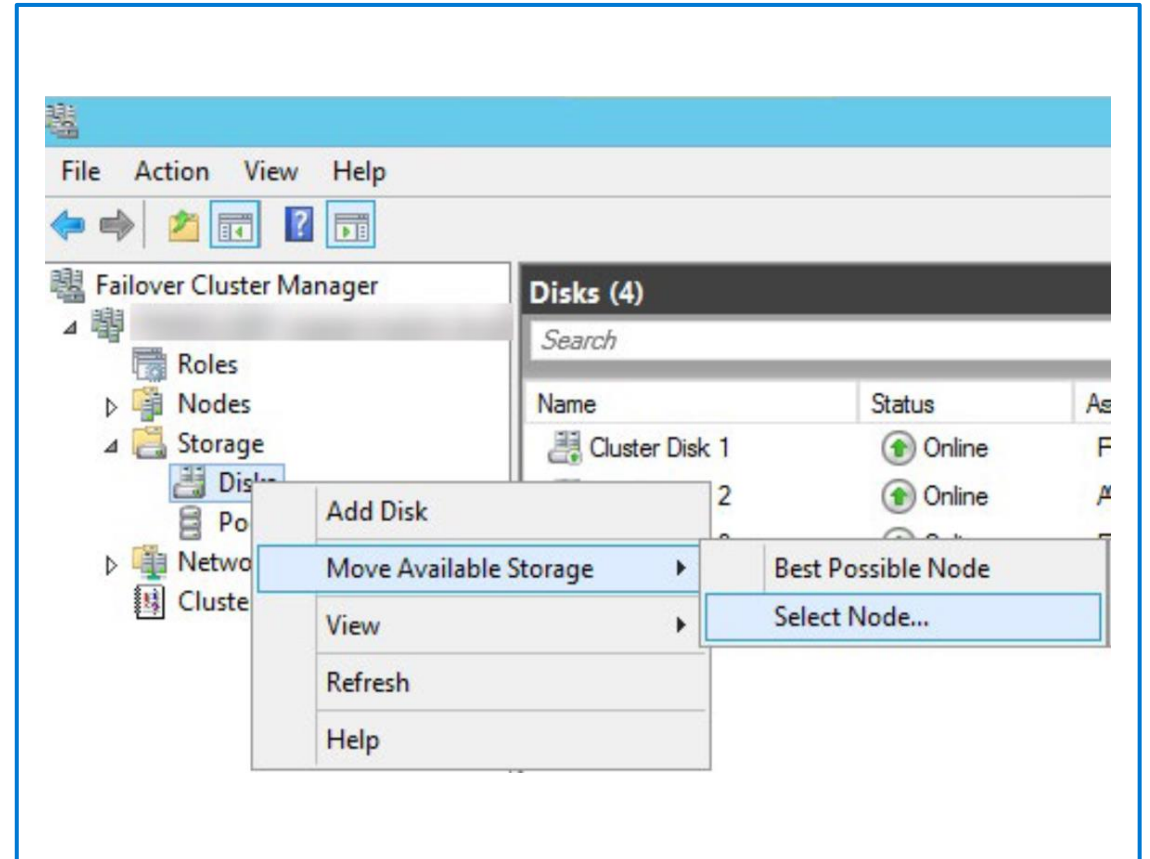
Test Cluster Group failover after creation

Test a Windows Server Failover Cluster

After the WSFC is created but before configuring FCIs or AGs, test that WSFC is working properly

Test the ability for all nodes to access shared storage

Select Node option to force the storage to fail over to other nodes in the cluster



Considerations for availability groups in Azure



Listener requires a **unique name** different from the WSFC, nodes, or anything else, as well as an **IP per subnet**

- Cannot reserve the listener's IP address in Azure



Deploying SQL Server VMs to multiple subnets is considered a best practice to avoid reliance on Azure Load Balancer or DNN for HADR traffic routing



AGs configurations require an underlying cluster, like an on-prem setup



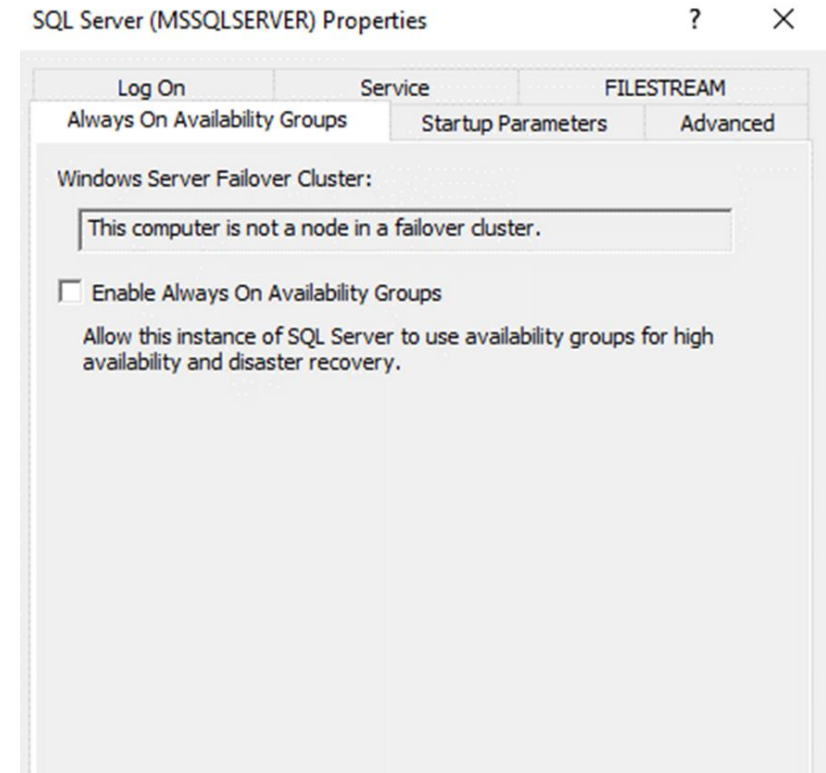
All VMs in AG should have the same storage configuration

Enable the Availability Groups feature

Cannot configure an AG without enabling the feature in **SQL Server Configuration Manager** or with `Enable-SqlAlwaysOn` in PowerShell

Must be done after configuring the WSFC

Requires instance **restart**



Create and Test an Availability Group

- ▶ The availability group creation process is the same as on premises, except for the process of creating the listener - Use SSMS, T-SQL, or PowerShell
- ▶ You may create the Azure Internal load balancer before or after creating the AG:
 - The ILB may be Basic or Standard—**Availability Zone deployments require Standard**
 - Create for the listener IP and the port used by SQL Server (not the AG endpoints)
 - Add the probe port to the IP address resource in the WSFC; will require a restart
- ▶ Test failover using SQL Server (T-SQL or SSMS)

Distributed Availability Groups



Like a regular AG, same creation process as on-premises



Requires creation of multiple WSFCs, AGs, and listeners



Requires an ILB for each WSFC/AG pair

Temporal tables



Temporal tables track and store data changes within a table



Tables must be converted to system-versioned to maintain this history

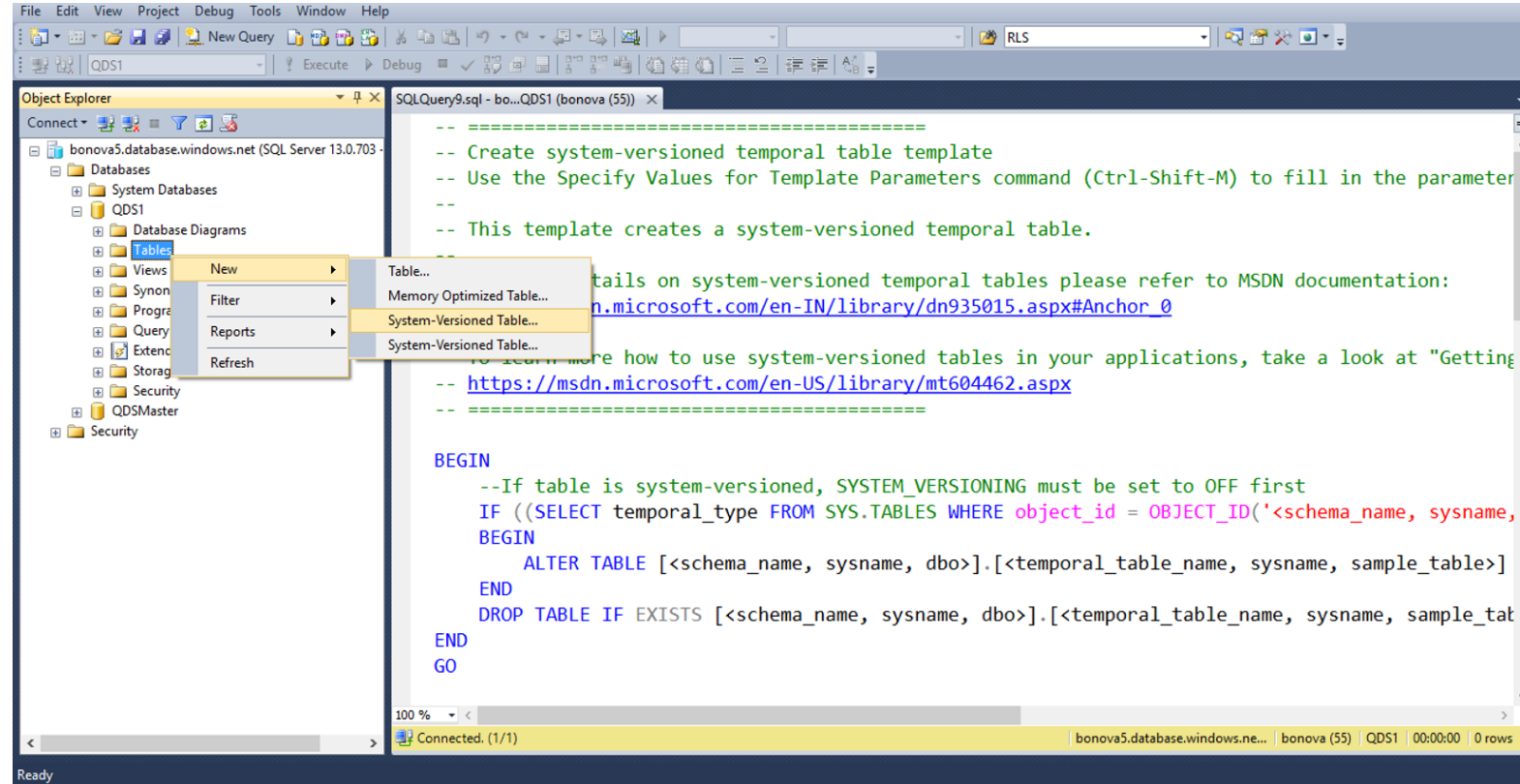


Additional system table is created to store version history



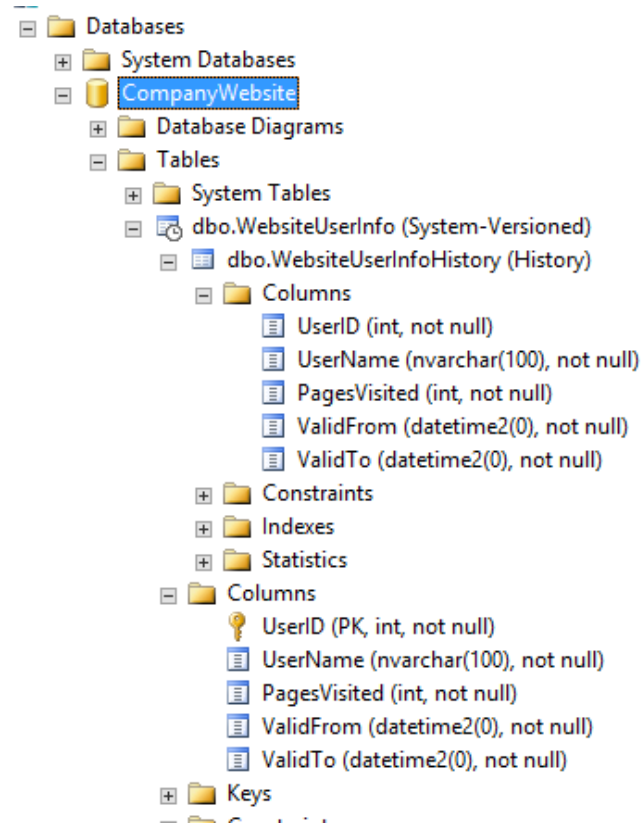
Allows for the manual T-SQL recovery of deleted or updated data if necessary, from the history table

Temporal table template

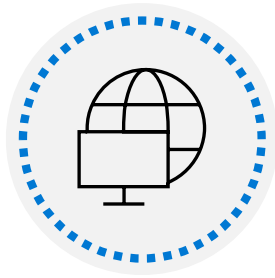


How to create a temporal table

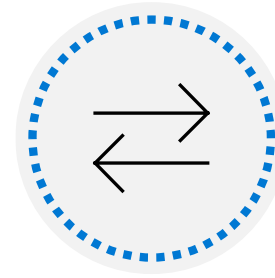
```
CREATE TABLE WebsiteUserInfo
(
    [UserID] int NOT NULL PRIMARY KEY CLUSTERED
    ,
    [UserName] nvarchar(100) NOT NULL ,
    [PagesVisited] int NOT NULL ,
    [ValidFrom] datetime2 (0) GENERATED ALWAYS
    AS ROW START,
    [ValidTo] datetime2 (0) GENERATED ALWAYS AS
    ROW END ,
    PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
) WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE =
dbo.WebsiteUserInfoHistory));
```



HADR Options for PaaS offering



Active Geo-Replication



Auto Failover Groups

Active Geo-Replication



Programmatically or manually failover primary databases to secondary regions during major disaster



Primary and secondary replicas are required to have the **same service tier and compute size**

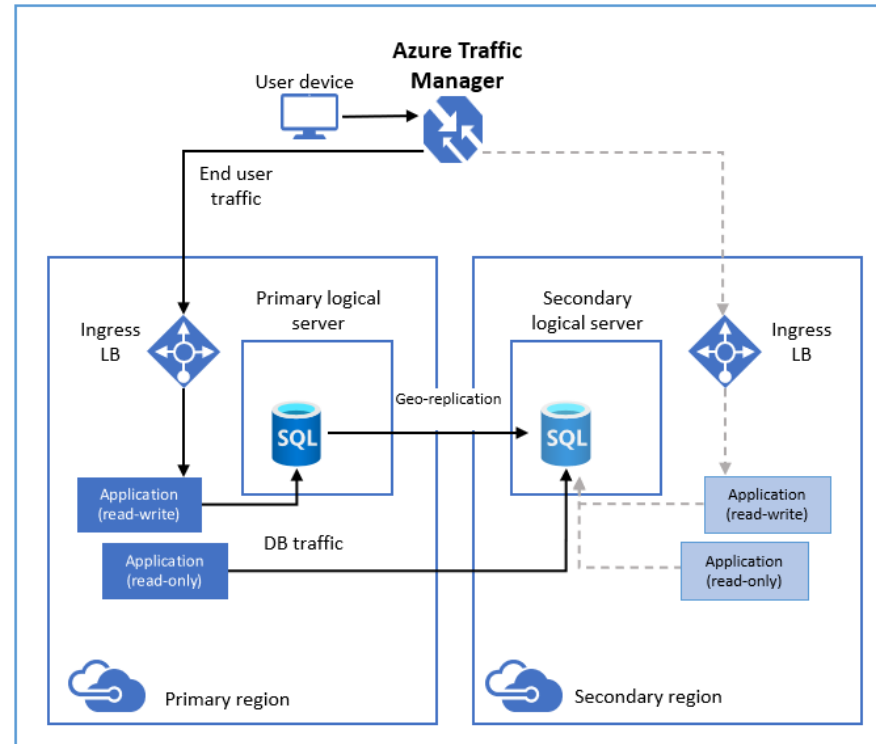


Cross subscription replication supported – configure a secondary replica on a different subscription than the primary database



Only supported on Azure SQL Database

Active Geo-Replication cont'd



Active Geo-Replication: Manual failover

After the secondary replica is created, you can manually fail over your secondary replica. The roles will switch with the secondary becoming the new primary, and the old primary the secondary.

[+ Create replica](#) [Refresh](#) [Feedback](#)

Geo replicas for your database are listed below. Geo replicas reside on a different logical server from the primary and protect against regional failures or prolonged data center outage. [Learn more](#)

Name ↑↓	Server ↑↓	Region ↑↓	Failover policy ↑↓	Pricing tier ↑↓	Replica state ↑↓	
▼ Primary						
AdventureWorksLT	dp300-lab-23075881	South Central US	None	General Purpose: Gen5, 2 vCores	Online	...
▼ Geo replicas						
AdventureWorksLT	dp300-lab-dr-23075881	East US		General Purpose: Gen5, 2 vCores	Readable	...

[Pin to dashboard](#)
[Stop replication](#)
[Forced failover](#)

Auto Failover Groups



Database is created automatically on the secondary through a process called **seeding**



It can contain one or more databases



Depending on the size of the database, the seeding process may take some time



If you have a tight RPO and can't afford much data loss, set the **GracePeriodWithDataLossHours** property to a higher value (default is 1-hour)



Supported on SQL Database and SQL Managed Instance

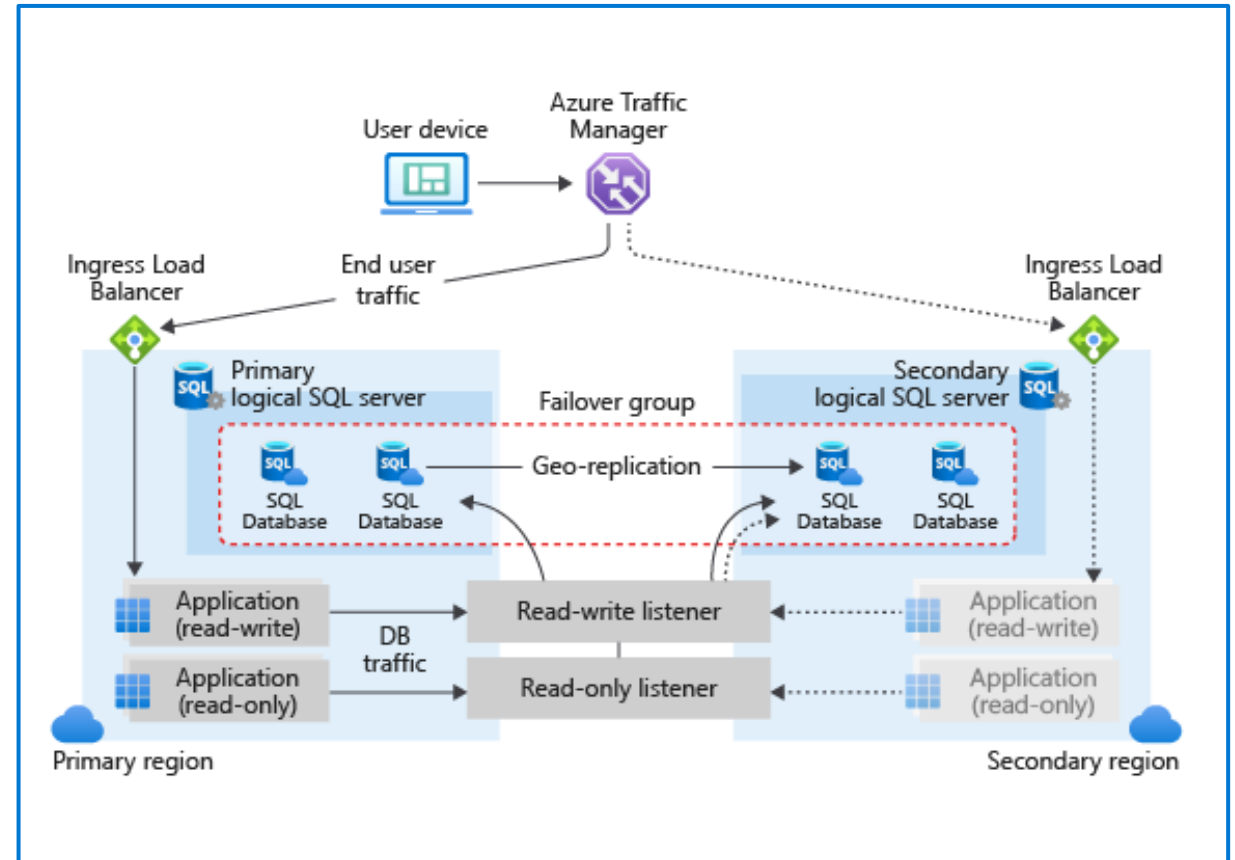
Auto Failover Groups – Azure SQL Database

There are two different kinds of listeners:

- For **read-write**
- For **read-only** traffic

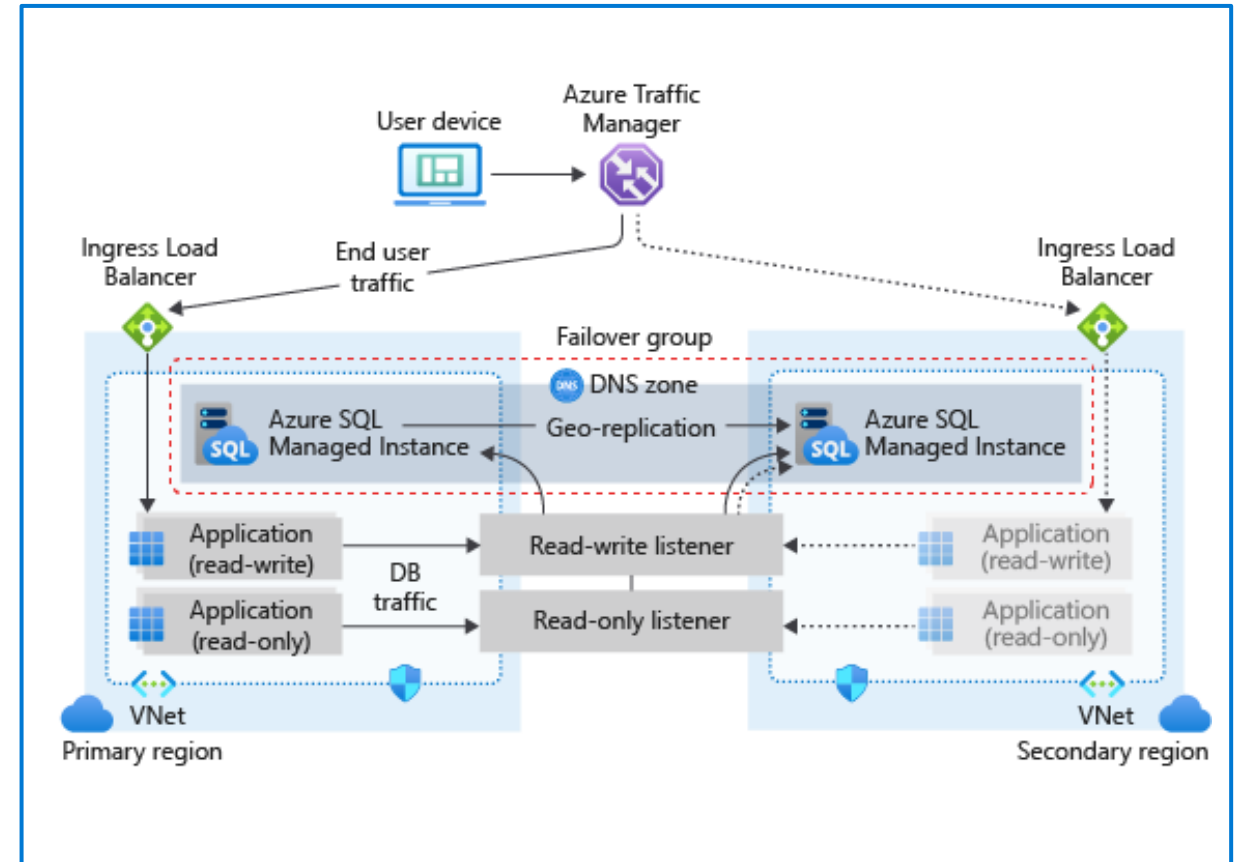
The secondary's database is created automatically through **seeding**

Behind the scenes in a failover, DNS is updated so clients will be able to point to the abstracted listener name



Auto Failover Groups – Azure SQL Managed Instance

- All user databases in the instance will be replicated to the secondary instance
- System databases like *master* and *msdb* will **not** be replicated
- Both the primary and secondary instances must be in the same DNS zone
- SQL Managed Instance failover groups in paired regions have better performance compared to unpaired regions
- Number of databases involved may affect seeding duration



Configure an auto-failover group for Azure SQL Database

Step 1:

Select **Failover groups** under the **Settings** pane, and then select **Add group** to create a new failover group

Home > Azure SQL resources > mySampleDatabase (mysqlserver/mySampleDatabase) > mysqlserver- Failover groups

mysqlserver - Failover groups
SQL server

Search (Ctrl+ /) « + Add group Refresh

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems

Settings

Quick start
Failover groups

Failover group are a SQL server feature designed to automatically manage repl

NAME	PRIMARY SERVER	SECOND
You have no group created		

Configure an auto-failover group cont'd

Step 2:

On the **Failover Group** page, enter or select the required values, and then select **Create**

Adding the database to the failover group will automatically start the geo-replication process

The screenshot displays the Azure portal interface for configuring a failover group. The breadcrumb navigation at the top reads: Home > Azure SQL resources > mySampleDatabase (mysqlserver/mySampleDatabase) > mysqlserver - Failover groups > Failover group > Databases.

Failover group configuration:

- Failover group name:** failovergrouptutorial (with a green checkmark and .database.windows.net suffix).
- Secondary server:** mysqlsecondary (East US).
- Read/Write failover policy:** Automatic.
- Read/Write grace period (hours):** 1 hours.
- Database within the group:** Select databases to add (highlighted with a red box).

Databases for failover group:

Select all (Selected/Eligible databases: 0/1)

Filter items...

NAME	ROLE	SECONDARY SERVER	STATUS
mySampleDatabase			Online

Summary:

- Databases on secondary (excluding ones in Elastic Pools): 0
- Elastic Pools on secondary server: 0

Monthly cost:

Auto Failover Groups vs. Geo-Replication

	Geo-replication	Failover groups
Automatic failover	No	Yes
Fail over multiple databases simultaneously	No	Yes
User must update connection string after failover	Yes	No
SQL Managed Instance support	No	Yes
Can be in same region as primary	Yes	No
Multiple replicas	Yes	No
Supports read-scale	Yes	Yes

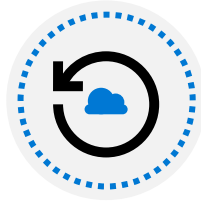
Instructor led labs: Configure geo-replication for Azure SQL Database

Enable geo-replication
Failover to a secondary region

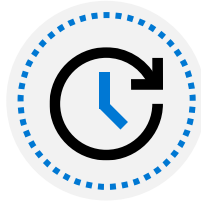
Back up and Restore Databases



Objectives



Backup and restore options for IaaS



Backup and restore options for PaaS

Backup and restore fundamentals for IaaS

SQL Server backup types:

- **Full** – backs up the entire whole database, but does not flush the transaction log
 - **Differential** – all changes since last full backup
 - **Transaction log** – backs up the transactions in the log since the transaction log backup
- Database must be in full or bulk-logged recovery model to generate a transaction log backup

Do not use Simple recovery model if you require point in time recovery

Full Azure Virtual Machine backup



Backs up full VM including SQL Server databases via VSS
Restores VM as a whole



Creates an application-consistent backup which will not 'break' SQL Server
Combining SQL Server backups with snapshots can potentially cause issues



May cause problems for some databases, always test the strategy



Ensure it can meet RTO and RPO



Causes a freeze/thaw of I/O of SQL Server log

Back up to Local disk or network share



Just like on premises; can perform native backups to disks attached to the VM or to a network share



Network share can be on Azure-based storage



If local disk, still need to ensure backups are periodically copied off of the VM for redundancy

Backup to and restore from URL

- ▶ Backup SQL Server databases directly to or restore from Azure blob storage
- ▶ Backup to URL can be used by **on premises** as well as IaaS instances
- ▶ Backup uses geo-replicated (GRS) storage accounts for another layer of data protection
- ▶ Requires authentication between the instance and Azure Storage
- ▶ SSMS, T-SQL, and PowerShell support

Automated IaaS VM Backups With The SQL Server Resource Provider

- Uses an Azure storage account as a target
- Azure manages the SQL Server backups (SQL VM)-
- Any IaaS VM that has SQL Server installed can use the **SQL Server resource provider**
- Can configure schedule, frequency, and retention
- **Choose one method to back up databases**

AUTOMATED BACKUP

Configure backups for databases in your virtual machine. All your SQL Server databases in this virtual machine will be backed up automatically per the settings you choose. If you decide to change settings via SQL Server Managed Backup in the future, the new settings will override the Automated Backup settings.

Automated backup

Retention period (days) 30

Storage account * **module7storage (Standard_RAGRS)**
<https://module7storage.blob.core.windows.net/>
[Select storage account](#)

Encryption

Backup system databases ⓘ

Configure backup schedule

Specify the schedule for full and log backups

FULL BACKUP SCHEDULE

Backup frequency

Take full backups every day at the specified start time

Backup start time (local VM time) ▼

Full backup time window (hours) 1

LOG BACKUP SCHEDULE

Backup frequency (minutes) 10

All your SQL Server databases in this virtual machine will be backed up automatically per the settings you choose. If you decide to change settings via SQL Server Managed Backup in the future, the new settings will override the Automated Backup settings.

Database backup and restore for Azure SQL Database

- Backups automatically generated: Full once a week, differential ever 12-24 hours, t-logs every 5-10 min
- Stored in **RA-GRS** blobs that are replicated to another datacenter
- Configured per database
- No user driven backup option
- Restore is per database
- It is possible to restore a deleted database

The screenshot displays the 'Configure policies' interface for an SQL server. It includes sections for 'Point In Time Restore Configuration' with a dropdown for 'Configure PITR backup retention' and a 'Days' field, and 'Long-term Retention Configurations' with checkboxes for 'Weekly LTR Backups', 'Monthly LTR Backups', and 'Yearly LTR Backups'. Each LTR option has a corresponding question about retention duration and a dropdown menu.

Configure policies
SQL server

Point In Time Restore Configuration

Configure PITR backup retention Days

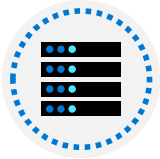
Long-term Retention Configurations

☐ Weekly LTR Backups ⓘ
How long would you like weekly backups to be kept?
 0 Week(s)

☐ Monthly LTR Backups ⓘ
How long would you like the first backup of each month to be kept?
 0 Week(s)

☐ Yearly LTR Backups ⓘ
Which weekly backup of the year would you like to retain?
 Week 1
How long would you like this annual backup to be kept?
 0 Week(s)

Database backup and restore for Azure SQL Managed Instance



Database backups automatically generated by default



Can restore to a point in time; like Azure SQL Database



Manually generate backups via backup to URL:

Requires credentials to access Azure Blob Storage

Can Restore too

Full backups can only generate a COPY_ONLY backup to maintain the log chain. i.e.

```
BACKUP DATABASE contoso
```

```
TO URL = 'https://myacc.blob.core.windows.net/testcontainer/contoso.bak' WITH COPY_ONLY
```



Backups generated cannot be used with Azure SQL Database

Instructor led labs: Backup to URL and Restore from URL

Create a credential

Backup to URL

Validate backup through Azure CLI and Storage Explorer

Restore from URL

Summary

Describe high availability and disaster recovery strategies:

- Understand Recovery time objective (RTO) and Recovery point objective (RPO)
- Understand the available HADR options for both IaaS and PaaS
- How to devise a HADR strategy

Back up and restore databases:

- Backup and restore options for IaaS
- Backup and restore options for PaaS

Explore IaaS and PaaS solutions for high availability and disaster recovery:

- The considerations for deploying a Windows Server Failover Cluster in Azure
- The considerations for deploying an Availability Group in Azure
- Temporal Tables
- Active-geo replication
- Auto-failover groups