ORACLE
Cloud Infrastructure

Deploying Microsoft SQL Server Always On Availability Groups

ORACLE WHITE PAPER | SEPTEMBER 2018

Disclaimer

Revision History

The following revisions have been made to this white paper since its initial publication:

Date	Revision
September 4, 2018	• Updated the route statement for the iSCSI target to be inclusive of current possible iSCSI targets versus individual routes per target. • Included procedures to enable common time synchronization between cluster node members.

You can find the most recent versions of the Oracle Cloud Infrastructure white papers at https://cloud.oracle.com/iaas/technical-resources.

Table of Contents

Target Audience

This white paper is intended for customers who are interested in deploying Microsoft SQL Server Always On availability groups in Oracle Cloud Infrastructure to meet enterprise-class database service availability and security requirements.

To perform a deployment as described in this paper, you should be familiar with the following processes and technologies:

- Windows Server and Active Directory administration using common administrative procedures, console snap-ins, or CLI tools
- Provisioning cloud infrastructure via Terraform, a popular and free lightweight deployment tool
- Executing PowerShell Cmdlets and scripts
- Windows Server Failover Clustering (WSFC) concepts and components
- SQL Server basic administration tasks, including creating a user database, navigating SQL Server Management Studio, and testing SQL Server client connections

You should also be familiar with the fundamentals of the Oracle Cloud Infrastructure. If this is the first time that you have used the platform, we recommend specifically the Launching Your First Linux Instance tutorial.

Introduction

Critical business applications demand a highly available and resilient data platform capable of automated host failure recovery. Microsoft SQL Server Always On availability groups in conjunction with Windows Server Failover Clustering (WSFC) deliver an enterprise-class solution for achieving stringent availability and data-protection requirements. Deployment to Oracle Cloud Infrastructure allows built-in cluster geo-redundancy for additional protection without additional administrative complexity.

This paper presents best-practice design elements and corresponding implementation procedures for deploying a Microsoft SQL Server Always On availability group cluster on Oracle Cloud Infrastructure. The material contained in this paper can be used as a reference for understanding optimal topology and configuration details during deployment planning. Additionally, the prescriptive installation and configuration steps can be used to deploy a proof-of-concept (POC) or production-ready environment.

Overview

Microsoft SQL Server Always On offers a diverse combination of operating modes and feature configurations to match specific workload requirements. Feature enhancements of SQL Server Enterprise Edition leverage the cluster-management capabilities of Windows Server to deliver the complete solution. This white paper covers the best-practice design configuration for a highly available service with the transactionally consistent failover capabilities typically required in a production implementation. Additionally, the reference architecture design adheres to security recommendations by implementing “least privilege required” managed domain accounts and other controls where applicable. Users who are interested in a streamlined temporary POC or non-production pilot implementation can select those options explicitly mentioned in this paper to reduce setup configuration complexity or reduce the infrastructure footprint.

The solution architecture topology consists of independent SQL Server instances on distinct Windows Server instances working together to host a discrete set of user databases, known as *availability databases*. At any point in time, a single set of primary read/write databases is colocated on a single instance. High-throughput, transactionally consistent background replication processes maintain secondary sets of non-writeable availability databases on independent servers. During an instance failure or planned maintenance activity, the status of availability databases and associated resources on a secondary instance might be automatically or manually promoted from standby to primary.

Note the following items about this architecture:

- SQL Server instances are independent and don’t share access to a storage subsystem or floating IP address. This independence eliminates the need for costly SAN or NAS systems and allows for geographically dispersed node topologies.
- Automatic failure detection and database failover can be triggered by common Windows host and SQL Server instance failures. Typically, logical data corruption, user database crashes, client connection blocking, and related activities on the database level don’t trigger a failover.
- The configuration described in this paper leverages transactionally consistent replication (also known as *synchronous-commit mode*) in an active-passive cluster configuration. SQL Server and Windows Server Failover Clustering (WSFC) offer numerous other feature configuration options such as heterogeneous replication modes, read-only secondary databases, symmetric cluster topologies, and so on, that might be relevant to your specific requirements. For more information, see the Microsoft documentation.

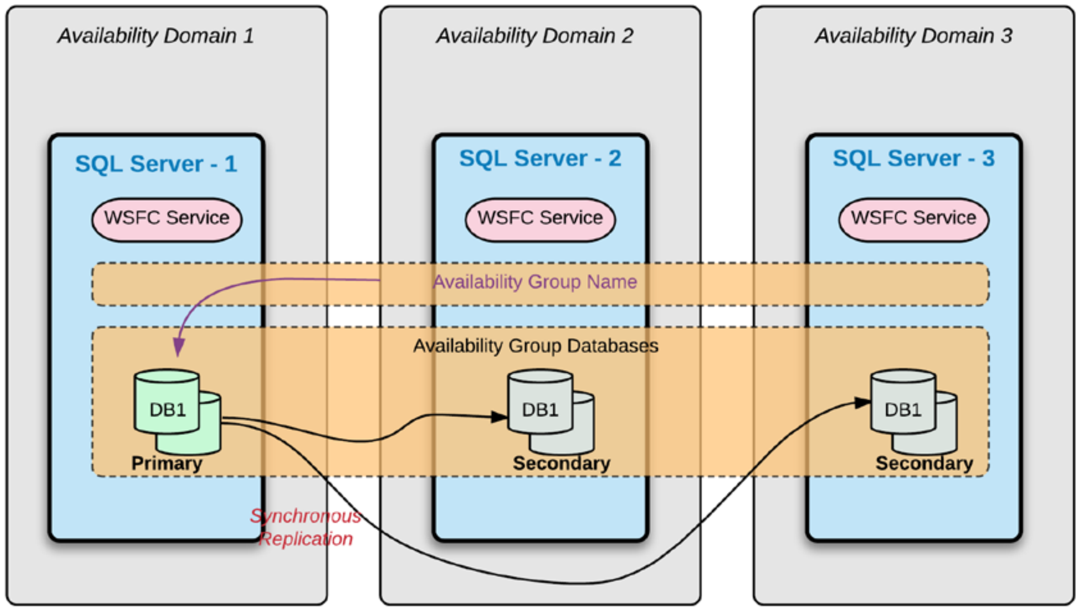
WSFC is leveraged to provide interserver coordination and resource management to support service high availability in a distributed environment. If a clustered Windows Server or SQL Server instance fails, the primary role of user databases and related services can be automatically or manually transferred to another available server. WSFC provides the following capabilities:

- Robust administrative tools and PowerShell commands to configure, review, and manage cluster deployments
- Active background health monitoring of cluster nodes and resources
- Automated host/instance failure recovery via resource failover and replication reconfiguration

Oracle Cloud Infrastructure further enhances the availability and resilience of this environment with several capabilities that are not typically available in traditional environments, such as the following ones:

- Deployment of individual cluster nodes in distinct *availability domains*, which are geographically separate physical data centers that are transparently connected by means of a high-speed network. This capability provides node isolation from many common failures related to building damage, power disruptions, or network ingress or egress slowdowns to the internet backbone.
- Agile deployment of new or replacement nodes, storage-capacity expansion, or instance resizing to quickly meet unpredictable loads without overprovisioning the environment.
- Capacity on demand to quickly replicate clusters for test validation of proposed changes. The ability to quickly and inexpensively replicate a production environment, schema, and data set might result in better testing with less risk of unintended consequences to the production environment.

The following diagram illustrates the SQL Server Always On availability group setup in an Oracle Cloud Infrastructure environment:



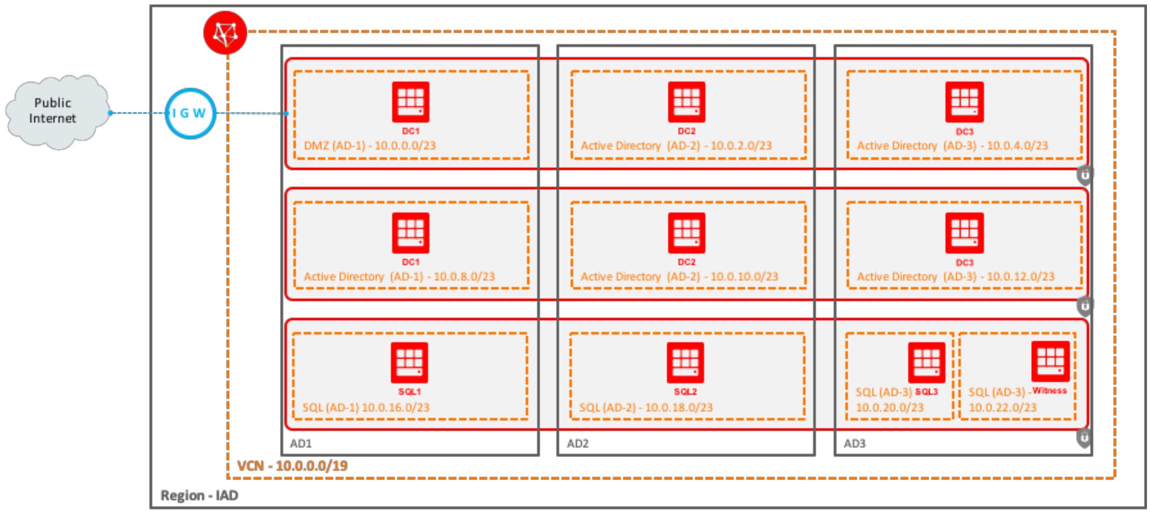
Methodology

This paper provides code examples for deploying infrastructure by using [Terraform] and also provides code examples for using PowerShell to configure Windows and SQL Server resources. Although most of the actions could be performed via GUI consoles and wizards, we strongly recommend that you become familiar with these tools and deployment methods. In particular, scripted and automated toolsets provide the following advantages:

- Predictable, repeatable, and consistent configurations
- Free from most errors related to subjective interpretations of GUI instructions
- Portable for quick implementation in non-production and test environments
- Configuration scripts for peer review and Security team signoff before making changes in production
- Typically, the fastest method to replace resources or restore configuration health during unplanned outages

Architecture

This white paper focuses on creating a SQL Server Always On availability group that spans all three availability domains (ADs) in a region, as illustrated in the following figure:


Public Internet
I G W
DC1
DMZ (AD-1) - 10.0.0.0/23
DC2
Active Directory (AD-2) - 10.0.2.0/23
DC3
Active Directory (AD-3) - 10.0.4.0/23
DC1
Active Directory (AD-1) - 10.0.8.0/23
DC2
Active Directory (AD-2) - 10.0.10.0/23
DC3
Active Directory (AD-3) - 10.0.12.0/23
SQL1
SQL (AD-1) 10.0.16.0/23
SQL2
SQL (AD-2) - 10.0.18.0/23
SQL (AD-3) SQL3
10.0.20.0/23
SQL (AD-3) - Witness
10.0.22.0/23
AD1
AD2
AD3
VCN - 10.0.0.0/19
Region - IAD

This paper references a Terraform template that implements the infrastructure configuration necessary to enable SQL Always On. The template's configuration is outlined in Appendix A.

The Terraform template deploys the following components:

- A compartment, SQLAlwaysOn, that contains all the resources for the environment.
- A virtual cloud network (VCN), which provides the network infrastructure for the solution. The network infrastructure can be configured to reside in an existing VCN, if necessary.
- Public and private subnets, which contain the Windows Server instances used for the solution. The following subnets are created:
    - Three public-facing DMZ subnets that contain a Windows Server instance to be configured as a bastion/jump host to access the private instances.
    - Three private administration subnets that each contain a Windows Server instance to be configured as the domain controllers.
    - Four private SQL Server subnets. Three of the subnets each contain a Windows Server "witness" instance to be configured as a node in the WSFC, a SQL Server Always On availability groups node, and a SQL Server system. The fourth subnet contains a Windows Server instance that acts as both the WSFC quorum share and the replica share (used by the SQL Always On availability group) for replication and synchronization.
- An internet gateway to provide internet access from the DMZ subnets.

- A security list with the appropriate security rules (covered in Appendix B) for each set of subnets (DMZ, Administration, SQL).
- Several block volumes used for both the SQL Server and the witness instances.
- Three private IP addresses for each SQL Server Windows instance.

Subnets

The following subnets are created by the Terraform template:

Name	CIDR	Availability Domain	Description
DMZ-AD1-IAD.sub	10.0.0.0/23	AD1	Used for bastion/jump hosts in AD1
DMZ-AD2-IAD.sub	10.0.2.0/23	AD2	Used for bastion/jump hosts in AD2
DMZ-AD3-IAD.sub	10.0.4.0/23	AD3	Used for bastion/jump hosts in AD3
Administration-AD1-IAD.sub	10.0.8.0/23	AD1	Hosts the Active Directory domain controllers in AD1
Administration-AD2-IAD.sub	10.0.10.0/23	AD2	Hosts the Active Directory domain controllers in AD2
Administration-AD3-IAD.sub	10.0.12.0/23	AD3	Hosts the Active Directory domain controllers in AD3
SQL-AD1-IAD.sub	10.0.16.0/23	AD1	Hosts the SQL Server instances in AD1
SQL-AD1-IAD.sub	10.0.16.0/23	AD1	Hosts the SQL Server instances in AD1
SQL-AD1-IAD.sub	10.0.16.0/23	AD1	Hosts the SQL Server instances in AD1
Witness-AD1-IAD.sub	10.0.16.0/23	AD1	Hosts the SQL Server instances in AD1

Instances

For the three-availability-domain architecture, the following instances are created: three bastion hosts, three Active Directory domain controllers, three SQL Server instances, and one cluster witness/quorum server, as outlined in the following table:

Name	Shape	Availability Domain	Subnet	FQDN
Bastion1	VM.Standard1.2	AD1	DMZ-AD1-iad.sub	Not applicable
Bastion2	VM.Standard1.2	AD2	DMZ-AD2-iad.sub	Not applicable

Name	Shape	Availability Domain	Subnet	FQDN
Bastion3	VM.Standard1.2	AD3	DMZ-AD3-iad.sub	Not applicable
DC1	VM.Standard1.2	AD1	Administration-AD1-iad.sub	DC1.SQLAlwaysOn.local
DC2	VM.Standard1.2	AD2	Administration-AD2-iad.sub	DC2.SQLAlwaysOn.local
DC3	VM.Standard1.2	AD3	Administration-AD3-iad.sub	DC3.SQLAlwaysOn.local
SQL1	VM.Standard1.8	AD1	SQL-AD1-iad.sub	SQL1.SQLAlwaysOn.local
SQL2	VM.Standard1.8	AD2	SQL-AD2-iad.sub	SQL2.SQLAlwaysOn.local
SQL3	VM.Standard1.8	AD3	SQL-AD2-iad.sub	SQL3.SQLAlwaysOn.local
Witness	VM.Standard1.2	AD3	Witness-AD3-iad.sub	Witness.SQLAlwaysOn.local

Storage on the SQL Server/WSFC Instances

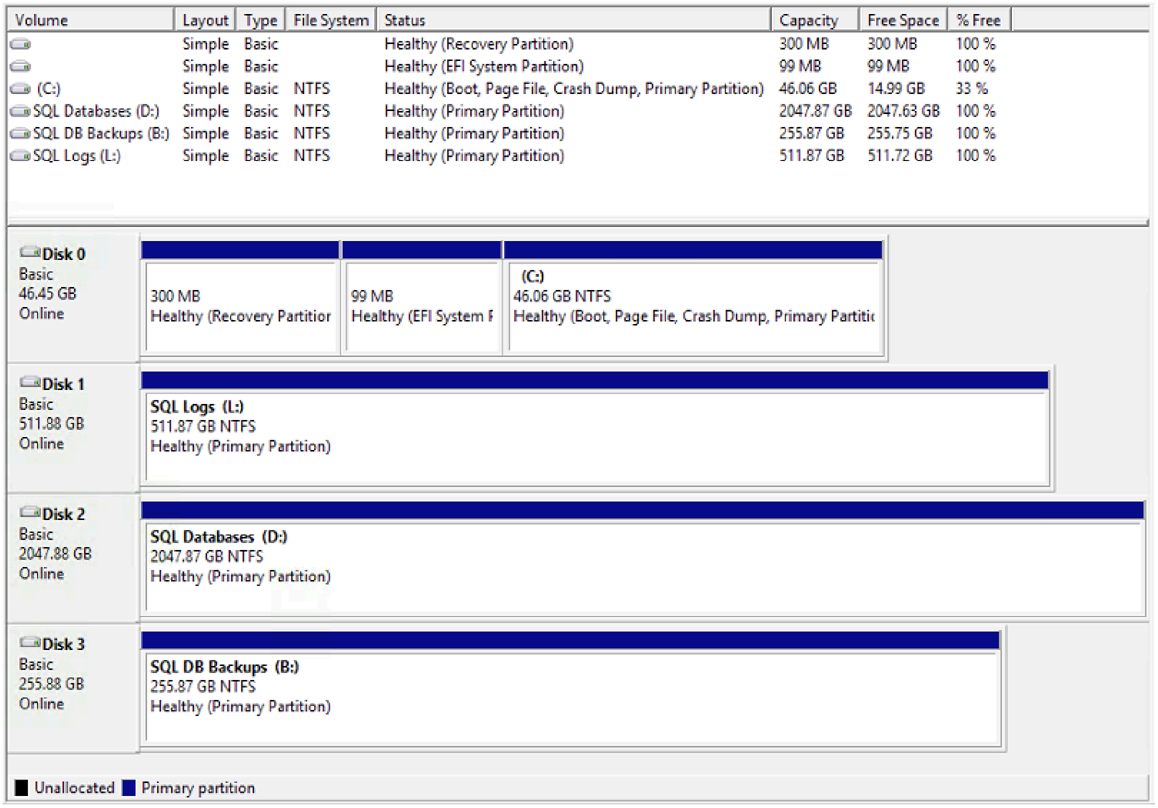
The amount of storage provided for each of the SQL Server/cluster instances varies depending on your workload. The Terraform template referenced in this paper deploy three different block volumes to each of these instances. (For more information about Oracle Cloud Infrastructure block volumes, see the Block Volume service documentation).

The three block volumes created for each of the SQL Server/cluster instances are as follows:

- 256-GB block volume for the SQL Server tempdb and backup files, to be mapped as `B:\`
- 2-TB block volume for the SQL Server database files, to be mapped as `D:\`
- 512-GB block volume for the SQL Server log files, to be mapped as `L:\`

Note: The block volume sizes are arbitrarily set for this paper. You can increase or decrease them as needed by modifying the `sql_db_size`, `sql_backup_size`, and `sql_log_size` Terraform variables (as outlined in the Appendix A). Different sizes were selected to make it easier to identify the volumes in each of the Windows instances when selecting which drive letter to assign to each. You can also use the Volume IQN to identify them. Ensure that you map the volumes to the drive letters correctly because they might be discovered and mounted in a different order in your environment as compared to this paper.

The following figure illustrates how the block volume partitions on each of the SQL Server/cluster instances appear in Disk Management after they have been attached, initialized, and formatted:



Volume	Layout	Type	File System	Status	Capacity	Free Space	% Free
	Simple	Basic		Healthy (Recovery Partition)	300 MB	300 MB	100 %
	Simple	Basic		Healthy (EFI System Partition)	99 MB	99 MB	100 %
(C:)	Simple	Basic	NTFS	Healthy (Boot, Page File, Crash Dump, Primary Partition)	46.06 GB	14.99 GB	33 %
SQL Databases (D:)	Simple	Basic	NTFS	Healthy (Primary Partition)	2047.87 GB	2047.63 GB	100 %
SQL DB Backups (B:)	Simple	Basic	NTFS	Healthy (Primary Partition)	255.87 GB	255.75 GB	100 %
SQL Logs (L:)	Simple	Basic	NTFS	Healthy (Primary Partition)	511.87 GB	511.72 GB	100 %

Disk 0
Basic
46.45 GB
Online

300 MB
Healthy (Recovery Partitior

99 MB
Healthy (EFI System F

(C:)
46.06 GB NTFS
Healthy (Boot, Page File, Crash Dump, Primary Partitic

Disk 1
Basic
511.88 GB
Online

SQL Logs (L:)
511.87 GB NTFS
Healthy (Primary Partition)

Disk 2
Basic
2047.88 GB
Online

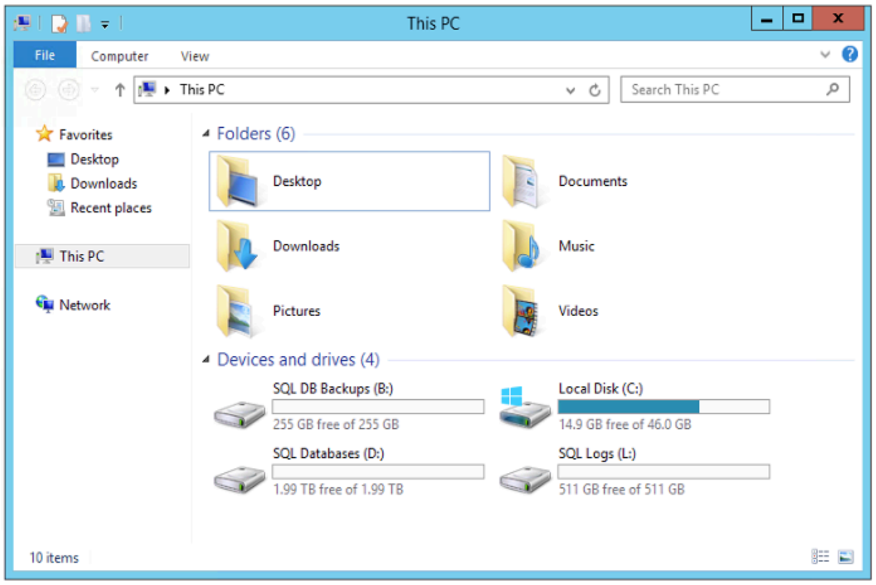
SQL Databases (D:)
2047.87 GB NTFS
Healthy (Primary Partition)

Disk 3
Basic
255.88 GB
Online

SQL DB Backups (B:)
255.87 GB NTFS
Healthy (Primary Partition)

■ Unallocated ■ Primary partition

The following figure illustrates how the block volume partitions on each of the SQL Server/cluster instances appear in Windows Explorer after they have been attached, initialized, and formatted:

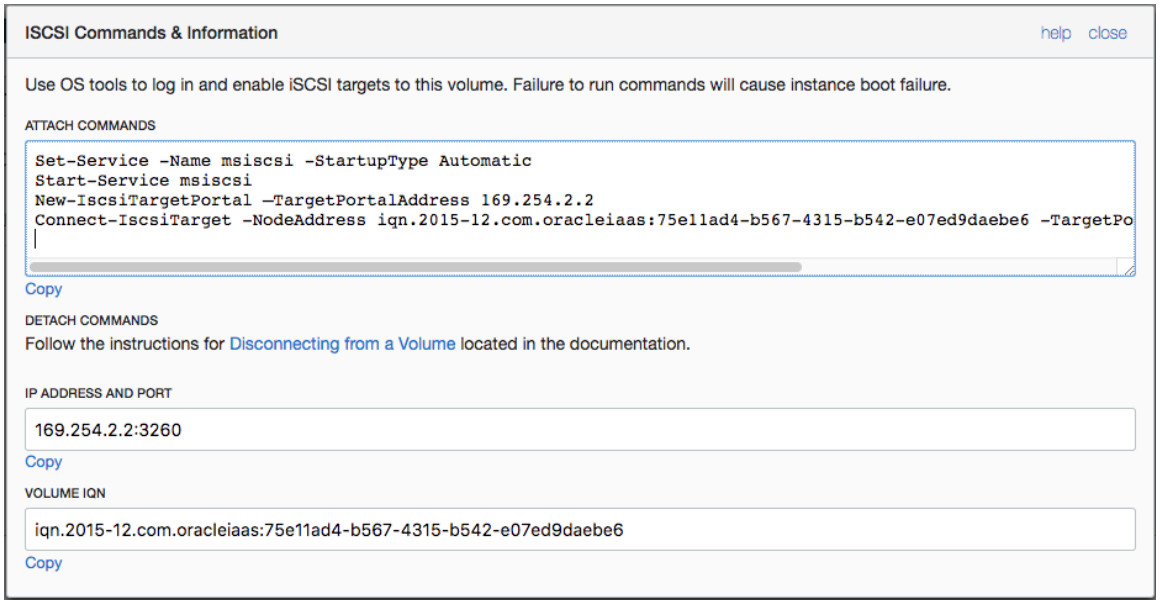


Another block volume is created for the witness instance. This volume is 256 GB in size and is formatted as the `S:\` drive. It is used to store the two shares used by the environment (the quorum share for the cluster and the replica share for the Always On availability group).

You must capture the iSCSI information from each of the block volumes because the iSCSI target portal addresses are required for each of the SQL Server/cluster and witness instances. The iSCSI target portal addresses are in the IP range 169.254.2.2 to 169.254.2.33. The following table lists the iSCSI target portal addresses used in this paper:

Instance Name	iSCSI Target Portal Address
SQL1	169.254.2.2 169.254.2.3 169.254.2.4
SQL2	169.254.2.2 169.254.2.3 169.254.2.4
SQL3	169.254.2.2 169.254.2.3 169.254.2.4
Witness	169.254.2.2

In the Console, you can find the iSCSI target portal address for each block volume by selecting the block volume and clicking the **iSCSI Commands & Information** button. You need the IP address listed in the **IP Address and Port** box, as shown in the following figure:



ISCSI Commands & Information

help close

Use OS tools to log in and enable iSCSI targets to this volume. Failure to run commands will cause instance boot failure.

ATTACH COMMANDS

```
Set-Service -Name msiscsi -StartupType Automatic
Start-Service msiscsi
New-IscsiTargetPortal –TargetPortalAddress 169.254.2.2
Connect-IscsiTarget -NodeAddress iqn.2015-12.com.oracleiaas:75e11ad4-b567-4315-b542-e07ed9daebe6 -TargetPo
```

Copy

DETACH COMMANDS

Follow the instructions for Disconnecting from a Volume located in the documentation.

IP ADDRESS AND PORT

169.254.2.2:3260

Copy

VOLUME IQN

iqn.2015-12.com.oracleiaas:75e11ad4-b567-4315-b542-e07ed9daebe6

Copy

If the addresses are different from the ones listed in the preceding table, capture them in the following table, replacing *x* with the appropriate value:

Instance Name	iSCSI Target Portal Address
SQL1	169.254.2.*x* 169.254.2.*x* 169.254.2.*x*
SQL2	169.254.2.x 169.254.2.*x* 169.254.2.*x*
SQL3	169.254.2.*x* 169.254.2.*x* 169.254.2.*x*
Witness	169.254.2.*x*

IP Addresses on the SQL Server/WSFC Instances

The Windows Server Failover Cluster (WSFC) and the SQL Server Always On availability group rely on their own TCP/IP addresses on each of the nodes. For WSFC and the availability groups to function, each of the SQL Server/cluster instances needs the following three IP addresses assigned to it:

- The first IP address is used as the primary IP address for the instance. This IP address is automatically created and assigned when the instance is provisioned.
- The second IP address is used for the WSFC instance. This IP address is created by the Terraform template after the instance has been provisioned.
- The third IP address is used for the SQL Always On availability group. This IP address is created by the Terraform template after the instance has been provisioned.

The IP addresses used for the environment defined in this paper are listed in the following table:

Instance Name	Primary IP Address	WSFC IP Address	Always On IP Address
Bastion1	10.0.0.2	Not applicable	Not applicable
Bastion2	10.0.2.2	Not applicable	Not applicable
Bastion3	10.0.4.2	Not applicable	Not applicable
DC1	10.0.8.2	Not applicable	Not applicable
DC2	10.0.10.2	Not applicable	Not applicable
DC3	10.0.12.2	Not applicable	Not applicable
SQL1	10.0.16.2	10.0.16.3	10.0.16.4
SQL2	10.0.18.2	10.0.18.3	10.0.18.4
SQL3	10.0.20.2	10.0.20.3	10.0.20.4
Witness	10.0.22.2	Not applicable	Not applicable

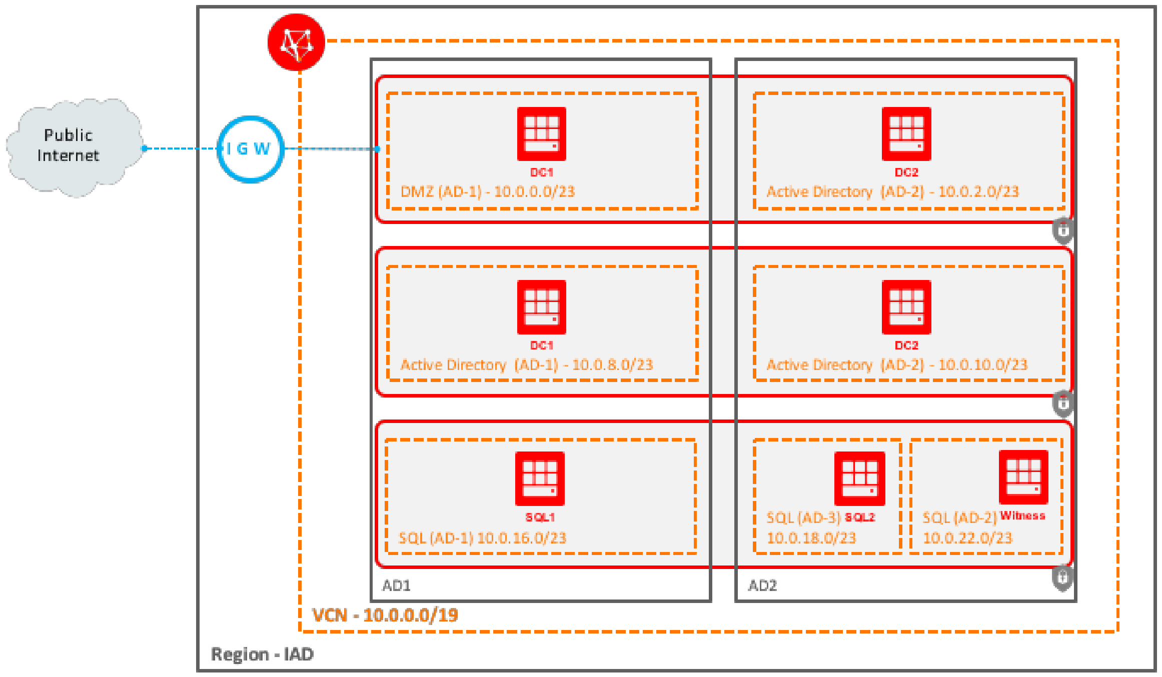
Execution of the Terraform template outputs the IP addresses that are assigned to each of the instances. If you change the template or the IP addresses that are assigned are different, you must update the commands provided in the deployment section of this paper. You can capture the IP addresses for your environment (if they are different) in the following table:

Instance Name	Primary IP Address	WSFC IP Address	Always On IP Address
Bastion1		Not applicable	Not applicable
Bastion2		Not applicable	Not applicable

Instance Name	Primary IP Address	WSFC IP Address	Always On IP Address
Bastion3		Not applicable	Not applicable
DC1		Not applicable	Not applicable
DC2		Not applicable	Not applicable
DC3		Not applicable	Not applicable
SQL1			
SQL2			
SQL3			
Witness		Not applicable	Not applicable

The Terraform also allows for the creation of a two-availability-domain model with some minor changes (covered in Appendix A).

The following figure shows the SQL Server Always On architecture with two availability domains:



Deploying the Environment

This section assumes that the configuration described in the preceding section has been created either by using the Terraform template or manually. After the base environment has been created, the information in this section provides instructions for deploying and configuring all remaining Microsoft components and features required to run SQL Server Always On availability groups on Oracle Cloud Infrastructure. These instructions assume that you are starting with a “clean slate” environment. If you are adding this environment to an existing environment, you must reconfigure the Terraform templates (or manually create the resources) and perform only the missing steps outlined in this section.

This section covers the following tasks:

1. Configure the Active Directory domain.
2. Configure time synchronization.
3. Configure the WSFC servers.
4. Install WSFC.
5. Install Microsoft SQL Server and Microsoft SQL Server Management Studio.
6. Create a SQL Server Always On availability group.
7. Test the WSFC and SQL Server Always On availability group.

Note: If you are using the Desktop Connection application (`mstsc.exe`) on the bastion host to connect to the servers, be sure to use the `<`*`server_name`*`>\opc` format to log on. For example, to connect to the DC1 server, use the following account: `DC1\opc`

Step 1: Configure the Active Directory Domain

Perform the following tasks on the Active Directory domain controllers that you have created.

Configure the DC1 Server

1. Use Remote Desktop to remotely connect to the DC1 server.

   Note: You must log in to each of the servers with the default password created for the instances automatically during the provisioning process. The default passwords are located on the Instance properties page in the Oracle Cloud Infrastructure Console.

2. Open a PowerShell command window as Administrator.

3. Install the Active Directory Domain Services role:

```
Install-WindowsFeature AD-Domain-Services –IncludeManagementTools
```

4. Set the password for the Administrator account:

   A. Run the Computer Management tool (`compmgmt.msc`).

   B. In the left pane, open **System Tools**, then **Local Users and Groups**, and then **Users**.

   C. Right-click the **Administrator** user and choose **Set Password**.

   D. Click **Proceed** and enter the new password.

5. Create the first domain in the Active Directory forest (you must specify a Safe Mode Administrator password):

```
Install-ADDSForest -CreateDnsDelegation:$false -DatabasePath
"C:\Windows\NTDS" -DomainMode "Win2012R2" -DomainName "sqlalwayson.local"
-DomainNetbiosName "SQLAlwaysOn" -ForestMode "Win2012R2" -InstallDns:$true
-LogPath "C:\Windows\NTDS" -NoRebootOnCompletion:$false -SysvolPath
"C:\Windows\SYSVOL" -Force:$true
```

   The server reboots when the task is completed.

6. When the server is back online, remotely connect to the DC1 server.

7. Add the OPC user to the Domain Admins group:

```
Add-ADGroupMember "Domain Admins" OPC
```

8. Create a new Active Directory group for all the SQL Server/cluster instances:

```
New-ADGroup –Name "SQL-Servers" -DisplayName "SQL Cluster Group" -
GroupCategory Security –GroupScope Global –Description "SQL Servers
Participating in AG Cluster"
```

Configure the DC2 and DC3 Servers

1. Install the Active Directory Domain Services role:

```
Install-WindowsFeature AD-Domain-Services –IncludeManagementTools
```

2. Set the DNS server to the DNS installed on the DC1 server:

```
Set-DNSClientServerAddress –interfaceIndex 12 –ServerAddresses
(“10.0.8.2")
```

3. Promote the server to a Domain Controller in the SQLAlwaysOn.local domain. You must enter the credentials for the Domain Admin account (`sqlalwayson\opc`) and specify a Safe Mode Administrator password.

```
Install-ADDSDomainController -InstallDns  -Credential (get-Credential
sqlalwayson\opc) -DomainName "sqlalwayson.local"
```

   The server reboots when the task is completed.

Create a Managed Service Account

You will use a Microsoft *managed service account* to run the SQL Server service. Doing so allows you to use an account that does not have a password associated with it directly but can be controlled by the computer accounts.

To create this account, run the following steps on either the DC2 server or the DC3 server, but not both:

1. Create a KDS root key, which is used to create the managed service account. Normally it can take up to 10 hours before this key becomes valid.

   A. For POC or test environments, run the following command:

   ```
   Add-KdsRootKey –EffectiveTime ((get-date).addhours(-10))
   ```

   B. For production environments, run the following command:

   ```
   Add-KdsRootKey -EffectiveImmediately
   ```

2. Create the new account and allow the members of the SQL-Servers group to use it:

```
New-ADServiceAccount –name SQLAGgMSA –DNSHostName
SQLAGMSA.SQLAlwaysOn.local –PrincipalsAllowedToRetrieveManagedPassword
"SQL-Servers"
```

3. Create a new group for the account. This group is used to access the replica share.

```
New-ADGroup –Name "SQL-gMSA" -DisplayName "SQL Service Account Group" -
GroupCategory Security –GroupScope Global –Description "SQL Service
Accounts for Replica share access"
```

4. Add the account to the newly created group:

```
Add-ADGroupMember "SQL-gMSA" SQLAGgMSA$
```

Step 2: Configure Time Synchronization

Clusters depend on consistent timestamps between servers. To ensure that all servers participating in the Always On environment have the same time, you must configure time synchronization.

Time can be synchronized either from your internal time servers or through the time service provided by Oracle Cloud Infrastructure. The process described here uses the Oracle Cloud Infrastructure time service as the source. If you want to use a different time source, ensure that the server is both resolvable and reachable from your environment, and replace the 169.254.169.254 address with your time server.

Perform the following steps on the each of the DC servers.

1. Open the Registry Editor.
2. Change the server type to NTP:
   A. Navigate to **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters**.
   B. Double-click **Type**.
   C. Change the value to **NTP**.
   D. Click **OK**.
3. Set AnnounceFlags to 5:
   A. Navigate to **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Config**.
   B. Double-click **AnnounceFlags**.
   C. Change the value to **5**.
   D. Click **OK**.

4. Enable the NTP server:
   A. Navigate to
      HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpServer
   B. Double-click **Enabled**.
   C. Change the value to **1**.
   D. Click **OK**.
5. Set the time sources:
   A. Navigate to
      HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters
   B. Double-click **NtpServer**.
   C. Change the value to **169.254.169.254,0x9**.
   D. Click **OK**.
6. Set the poll interval:
   A. Navigate to
      HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient
   B. Double-click **SpecialPollInterval**.
   C. Change the value to an interval in seconds, for example, 900 (15 minutes).
   D. Click **OK**.
7. Configure correction settings:
   A. Navigate to
      HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Config
   B. Double-click **MaxPosPhaseCorrection**.
   C. Change the value to an interval in seconds, for example, 1800 (30 minutes).
   D. Click **OK**.

E. Double-click **MaxNegPhaseCorrection**.

F. Change the value to an interval in seconds, for example, 1800 (30 minutes).

G. Click **OK**.

8. Restart the time service. From the command prompt (CMD) or PowerShell, run the following command:

```
net stop w32time && net start w32time
```

9. Test the connection to the NTP server. From the command prompt (CMD) or PowerShell, run the following command:

```
w32tm /query /peers
```

You should enable time synchronization for cluster members as well. Depending on your environment requirements, the time synchronization source can be either the Oracle Cloud Infrastructure time source (169.254.169.254) or the DC servers configured as part of this process.

Step 3: Configure the SQL Server/WSFC Servers

Run the following steps on the SQL1, SQL2, SQL3, and Witness servers:

1. Set the DNS server to the DNS installed on the DC1 server:

```
Set-DNSClientServerAddress –interfaceIndex 12 –ServerAddresses
("10.0.8.2")
```

2. Join the SQLAlwaysOn.local domain:

```
Add-Computer -domain "sqlalwayson.local" -Credential (get-Credential
sqlalwayson\opc) -restart
```

The servers reboot after the domain is joined.

After the SQL Server servers are back online, run the following steps on them, but *not* on the Witness server:

1. Install the Windows Server Failover Cluster (WSFC) feature:

```
Install-WindowsFeature -Name Failover-Clustering –IncludeManagementTools -
restart
```

The server reboots after the feature installation is complete.

The WSFC feature uses the 169.254.*x.x* network for some of its internal communication. This network is also used by Oracle Cloud Infrastructure to expose the iSCSI block volume devices. To ensure that the block volumes can be connected to from the Windows Server instances, you must create static routes.

2. Create static IP routes to expose the iSCSI target portals. The following routes allow the three volumes configured for each server to be visible to the iSCSI Initiator Service, and account for any additional volumes that must be added in the future.

   On SQL1:

   ```
   New-NetRoute –DestinationPrefix "169.254.2.0/24" –InterfaceIndex 12 –NextHop
   10.0.16.2 -publish yes
   ```

   On SQL2:

   ```
   New-NetRoute –DestinationPrefix "169.254.2.0/24" –InterfaceIndex 12 –NextHop
   10.0.18.2 -publish yes
   ```

   On SQL3:

   ```
   New-NetRoute –DestinationPrefix "169.254.2.0/24" –InterfaceIndex 12 –NextHop
   10.0.20.2 -publish yes
   ```

3. Create the new iSCSI target portals:

   ```
   New-IscsiTargetPortal -TargetPortalAddress 169.254.2.2
   New-IscsiTargetPortal -TargetPortalAddress 169.254.2.3
   New-IscsiTargetPortal -TargetPortalAddress 169.254.2.4
   ```

4. Connect all the iSCSI disks:

   ```
   Get-IscsiTarget | Connect-IscsiTarget -IsPersistent $true
   ```

5. Verify that the disks have been discovered and are online by running the following command:

   ```
   Get-Disk
   ```

6. If any of the disks appear as Offline, run the following commands to bring them online:

   ```
   Set-Disk -Number 1 -IsOffline $False
   Set-Disk -Number 2 -IsOffline $False
   Set-Disk -Number 3 -IsOffline $False
   ```

7. Initialize the newly discovered disks:

```
Initialize-Disk 1
Initialize-Disk 2
Initialize-Disk 3
```

8. Create new partitions (one on each of the drives). Ensure that the partitions match the drive letter as shown in the “Storage on the SQL Server/WSFC Instances” section.

```
New-Partition -DiskNumber 1 -DriveLetter B -UseMaximumSize
New-Partition -DiskNumber 2 -DriveLetter D -UseMaximumSize
New-Partition -DiskNumber 3 -DriveLetter L –UseMaximumSize
```

Note: The preceding commands assume that disk 1 is the 256-GB block volume mapped to the `B:\` drive, disk 2 is the 2-TB block volume mapped to the `D:\` drive, and disk 3 is the 512-GB block volume mapped to the `L:\` drive.

9. Format the newly created volumes and assign the appropriate drive letter to them (you must manually approve the formatting of each of the volumes):

```
Format-Volume -DriveLetter B -FileSystem NTFS -NewFileSystemLabel "SQL DB Backups"
Format-Volume -DriveLetter D -FileSystem NTFS -NewFileSystemLabel "SQL Databases"
Format-Volume -DriveLetter L -FileSystem NTFS -NewFileSystemLabel "SQL DB Logs"
```

Note: Although the `B:` drive is normally reserved for a physical floppy drive, the systems in Oracle Cloud Infrastructure don’t associate anything with the `B:` drive by default.

10. Create the firewall rule to allow SQL access (port 1433):

```
New-NetFirewallRule -Name "SQL Server Rule (in)" -Description "SQL Server
Rule" -DisplayName "SQL Rule" -Enabled:True -Profile Domain,Public,Private
-Direction Inbound -Action Allow -Protocol TCP -LocalPort 1433
```

11. Create the firewall rule to allow SQL Server access for the Always On availability groups endpoints (port 5022):

```
New-NetFirewallRule -Name "SQL Server (port 5022) Rule (in)" -Description
"SQL Server (port 5022) Rule" -DisplayName "SQL (port 5022) Rule" -
Enabled:True -Profile Domain,Public,Private -Direction Inbound -Action
Allow -Protocol TCP -LocalPort 5022
```

Run the following steps on the Witness server:

1. Create the new iSCSI target portal:

```
New-IscsiTargetPortal -TargetPortalAddress 169.254.2.2
```

2. Connect the iSCSI disk:

```
Get-IscsiTarget | Connect-IscsiTarget -IsPersistent $true
```

3. Verify that the disk has been discovered and is online by running the following command:

```
Get-Disk
```

4. If the disk appears as offline, run the following commands to bring it online:

```
Set-Disk -Number 1 -IsOffline $False
```

5. Initialize the newly discovered disk:

```
Initialize-Disk 1
```

6. Create new partitions:

```
New-Partition -DiskNumber 1 -DriveLetter S -UseMaximumSize
```

7. Format the newly created volume:

```
Format-Volume -DriveLetter S -FileSystem NTFS
```

8. Create a new directory for the cluster quorum:

```
New-Item S:\Quorum -Type directory
```

9. Create a share and assign the correct permissions to the share to allow the cluster instances to connect:

```
New-SmbShare -Name "Quorum" -Path "S:\Quorum" -FullAccess
"sqlalwayson\SQL-Servers","sqlalwayson\Domain Admins"
```

10. Create a new directory for the Always On availability groups replicas:

```
New-Item S:\Replica -Type directory
```

11. Create a share:

```
New-SmbShare -Name "Replica" -Path "S:\Replica" -FullAccess
"sqlalwayson\SQL-gMSA"
```

12. Assign the correct permissions to the share to allow the Always On managed service account to connect:

    A. Open Windows Explorer.

    B. Navigate to **S:\Replica**.

C. Right-click the **Replica** folder, choose **Share with**, and then choose **Specific people**.

D. In the text field, enter **SQL-gMSA** and then click **Add**.

E. From the drop-down menu to the right of **SQL-gMSA**, choose **Read/Write**.

F. Click **Share**.

G. Click **Done**.

13. Add the newly created SQL Server computer accounts to the group by running the following command on the DC1 server:

```
Add-ADGroupMember "SQL-Servers" SQL1$, SQL2$, SQL3$
```

Step 4: Install WSFC

Run the following steps on any of the SQL Server instances:

1. Create the WSFC cluster:

```
New-Cluster -Name Cluster1 -Node SQL1,SQL2,SQL3 -StaticAddress
10.0.16.3,10.0.18.3,10.0.20.3 -NoStorage
```

2. Configure the quorum cluster:

```
Set-ClusterQuorum -NodeAndFileShareMajority \\Witness\Quorum
```

Step 5: Install Microsoft SQL Server and Microsoft SQL Server Management Studio

Note: These installation steps are for SQL Server 2017 Enterprise. You can use any version of SQL Server that supports SQL Server Always On availability groups, but the installation steps might vary slightly.

You can install Microsoft SQL Server on each of the SQL Server instances (SQL1, SQL2, and SQL3). The installation needed for Always On availability groups is the default standalone installation. You should not need to make any changes during this step; any necessary changes will be done in the next step.

If you are using a configuration similar to the one outlined in this paper, you can use the configuration file on the Terraform GitHub site by following these steps:

1. Save the file as `ConfigurationFile.ini` and copy it to each of the SQL Server instances (save it in the `C:\Temp` folder, for example).
2. Execute the following command from the location of the SQL Server `setup.exe` application:
   ```
   .\setup.exe /configurationfile=C:\Temp\ConfigurationFile.ini
   ```
3. After you execute the command, choose the version to install, accept the license agreement, and click through the wizard. The required configuration is prepopulated in the wizard from the configuration file.
4. Install the Microsoft SQL Server Management Studio (SSMS) on each of the SQL Server instances.

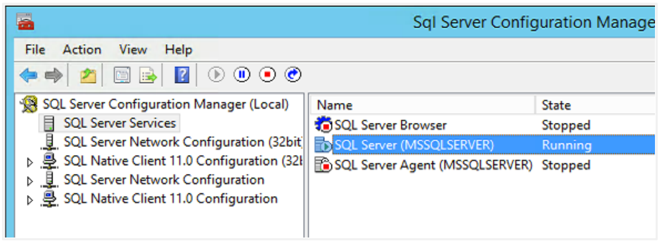
Step 6: Create a SQL Server Always On Availability Group

Now you can create the SQL Server Always On availability group.

Configure the Service Account

By default, the SQL Server service runs as a built-in account (`NTService\MSSQLSERVER`). For SQL Server Always On to function, you need to change it to use the managed service account that you created by following these steps on all the SQL Server instances:

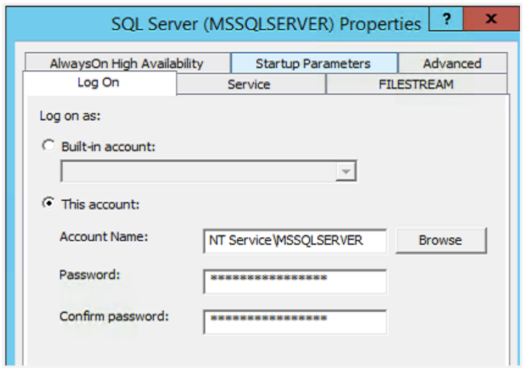
1. Launch the SQL Server Configuration Manager.
2. In the left navigation pane, select **SQL Server Services**.


Sql Server Configuration Manage
File
Action
View
Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32l
SQL Server Network Configuration
SQL Native Client 11.0 Configuration

Name	State
SQL Server Browser	Stopped
SQL Server (MSSQLSERVER)	Running
SQL Server Agent (MSSQLSERVER)	Stopped

3. Right-click the **SQL Server (MSSQLSERVER)** option and select **Properties**.

4. In the properties dialog box, click **Browse** next to **Account Name**.


SQL Server (MSSQLSERVER) Properties
AlwaysOn High Availability
Startup Parameters
Advanced
Log On
Service
FILESTREAM
Browse

Log on as:

Built-in account:

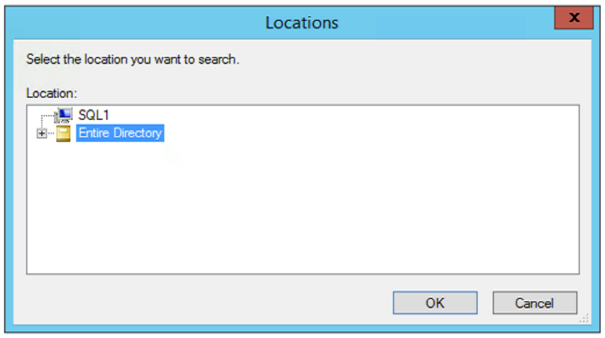
(•) This account:

Account Name: NT Service\MSSQLSERVER

Password: ****************

Confirm password: ****************

5. Change the location by clicking the **Locations** button, selecting **Entire Directory**, and then clicking **OK**.

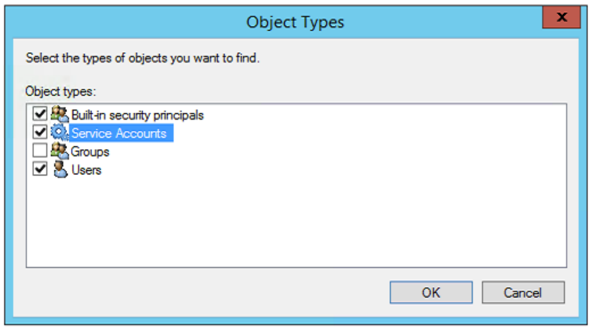


Select the location you want to search.

Location:

- SQL1
- Entire Directory

6. Enable the adding of service accounts by clicking on the **Object Types** button, selecting **Service Accounts**, and then clicking **OK**.

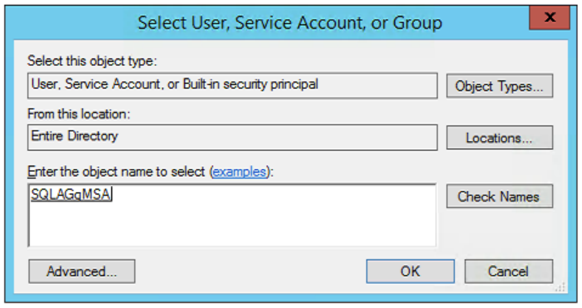


Select the types of objects you want to find.

Object types:

- ☑ Built-in security principals
- ☑ Service Accounts
- ☐ Groups
- ☑ Users

7. In the **Enter the object name to select** text box, type the name of the service account (**SQLAGgMSA**), and then click **OK**.



Select User, Service Account, or Group

Select this object type:
User, Service Account, or Built-in security principal

From this location:
Entire Directory

Enter the object name to select (examples):
SQLAGgMSA

8. Click **OK** to apply the changes and close the Properties dialog box.

   A warning appears, stating that this action causes the service to restart.

9. Click **Yes** to complete the configuration and restart the service.

10. Repeat these steps on all the SQL Server instances in the environment.

Create the SQL Server Always On Availability Group

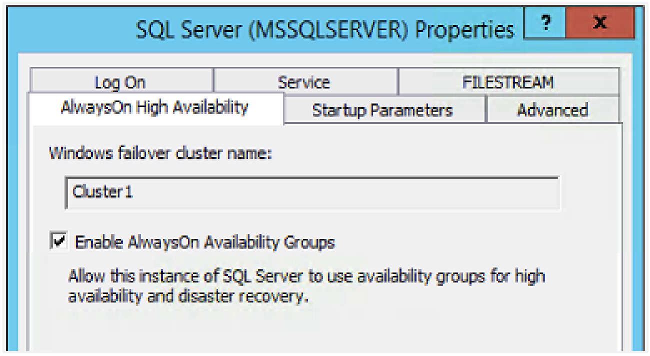
This section walks through the required steps to create the SQL Server Always On availability group.

Enable the SQL Server AlwaysOn High Availability Option

Run the following command on each of the SQL Server instances, substituting the name of the SQL Server instance (SQL1, SQL2, or SQL3) for *<SQL_server>*.

```
Enable-SqlAlwaysOn -ServerInstance <SQL_server> -force
```

You can verify that the feature was enabled by viewing the **AlwaysOn High Availability** tab in the SQL Server Service Properties dialog box.



SQL Server (MSSQLSERVER) Properties

Windows failover cluster name:
Cluster1

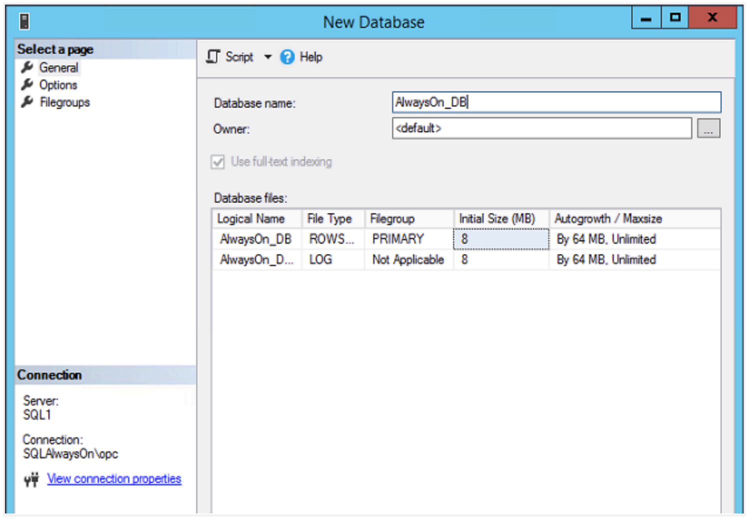
☑ Enable AlwaysOn Availability Groups

Allow this instance of SQL Server to use availability groups for high availability and disaster recovery.

Create or Attach a Database

Either create a test database or attach an existing database. The following steps create a test database named AlwaysOn_DB.

1. Connect to the first instance in the cluster, SQL1.
2. Run the Microsoft SQL Server Management Studio (SSMS) application.
3. In the **Connect to Server** dialog box, connect to the SQL1 instance.
4. In the Object Explorer pane of SSMS, right-click **Databases** and select **New Database**.
5. On the **General** page of the **New Database** dialog box, enter `AlwaysOn_DB` as the database name.



Database name: AlwaysOn_DB

Owner: <default>

Use full-text indexing

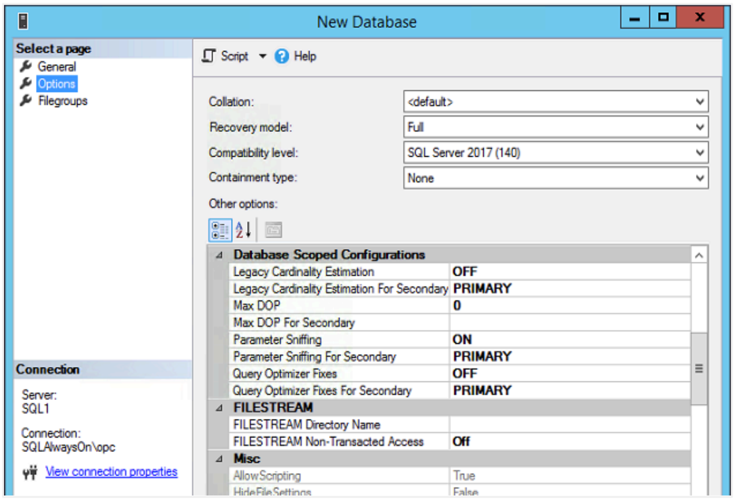
Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize
AlwaysOn_DB	ROWS...	PRIMARY	8	By 64 MB, Unlimited
AlwaysOn_D...	LOG	Not Applicable	8	By 64 MB, Unlimited

Server:
SQL1

Connection:
SQLAlwaysOn\opc

6. Click the **Options** page, and ensure that **Recovery model** is set to **Full**.



Server:
SQL1

Connection:
SQLAlwaysOn\opc

Collation: <default>
Recovery model: Full
Compatibility level: SQL Server 2017 (140)
Containment type: None

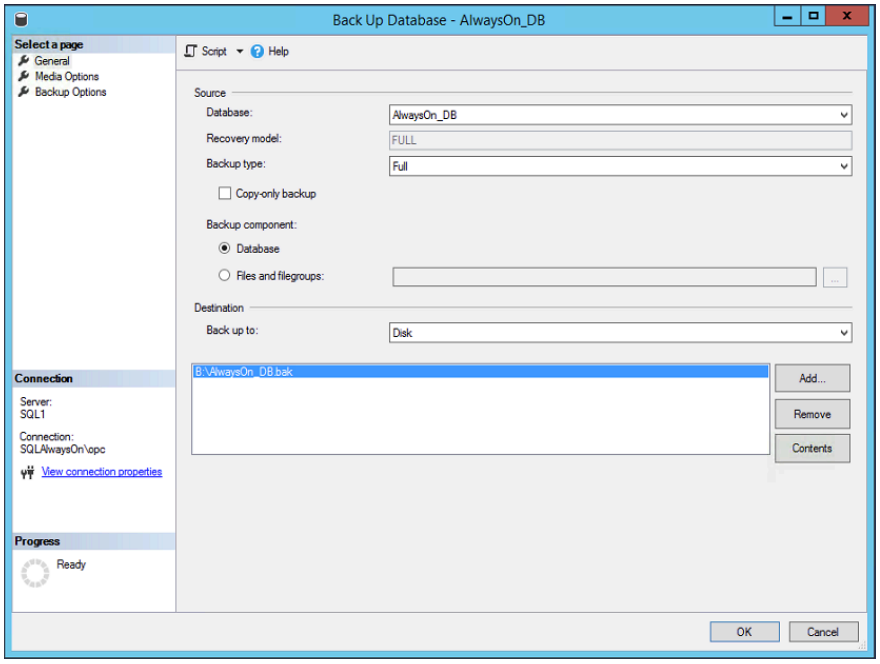
Other options:

Database Scoped Configurations	
Legacy Cardinality Estimation	OFF
Legacy Cardinality Estimation For Secondary	PRIMARY
Max DOP	0
Max DOP For Secondary	
Parameter Sniffing	ON
Parameter Sniffing For Secondary	PRIMARY
Query Optimizer Fixes	OFF
Query Optimizer Fixes For Secondary	PRIMARY
FILESTREAM	
FILESTREAM Directory Name	
FILESTREAM Non-Transacted Access	Off
Misc	
AllowScripting	True
HideFileSettings	False

7. Click **OK** to create the database.

8. Create a full backup of the database by right-clicking the newly created database in the Object Explorer pane and selecting **Tasks** and then selecting **Back Up**.

9. In the **Back Up Database** dialog box, click **OK** to complete the backup.



Server:
SQL1

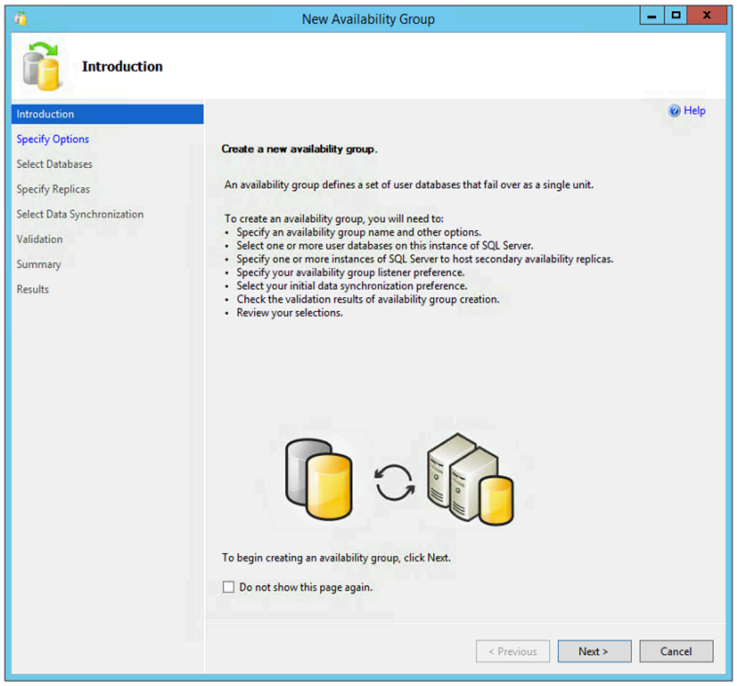
Connection:
SQLAlwaysOn\opc

Source
Database: AlwaysOn_DB
Recovery model: FULL
Backup type: Full
Copy-only backup
Backup component:
[✓ Database]
Files and filegroups:

Destination
Back up to: Disk
B:\AlwaysOn_DB.bak

Create the Availability Group

1. In the Object Explorer pane of Microsoft SQL Server Management Studio, right-click **AlwaysOn High Availability** and select **New Availability Group Wizard**.



Create a new availability group.

An availability group defines a set of user databases that fail over as a single unit.

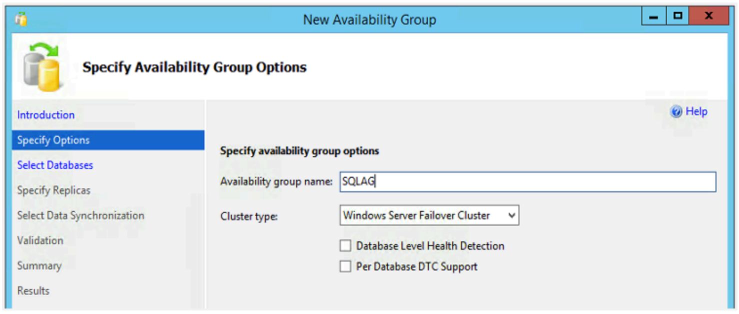
To create an availability group, you will need to:
- Specify an availability group name and other options.
- Select one or more user databases on this instance of SQL Server.
- Specify one or more instances of SQL Server to host secondary availability replicas.
- Specify your availability group listener preference.
- Select your initial data synchronization preference.
- Check the validation results of availability group creation.
- Review your selections.

To begin creating an availability group, click Next.

☐ Do not show this page again.

2. Click **Next**.

3. On the **Specify Availability Group Options** page, type `SQLAG` as the availability group name, and then click **Next**.



Specify availability group options

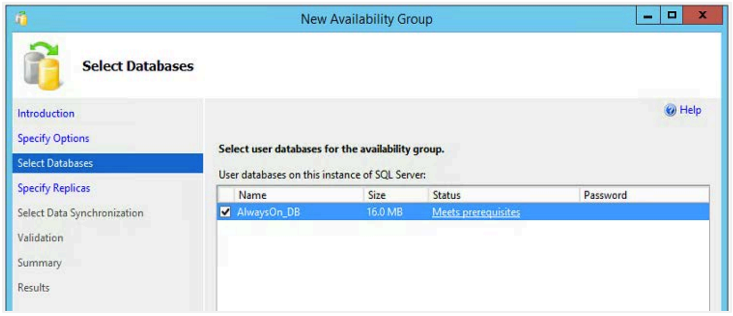
Availability group name: SQLAG

Cluster type: Windows Server Failover Cluster

☐ Database Level Health Detection

☐ Per Database DTC Support

4. On the **Select Databases** page, ensure that the newly created database (AlwaysOn_DB) shows that it meets the prerequisites. If it does, select it and click **Next**.



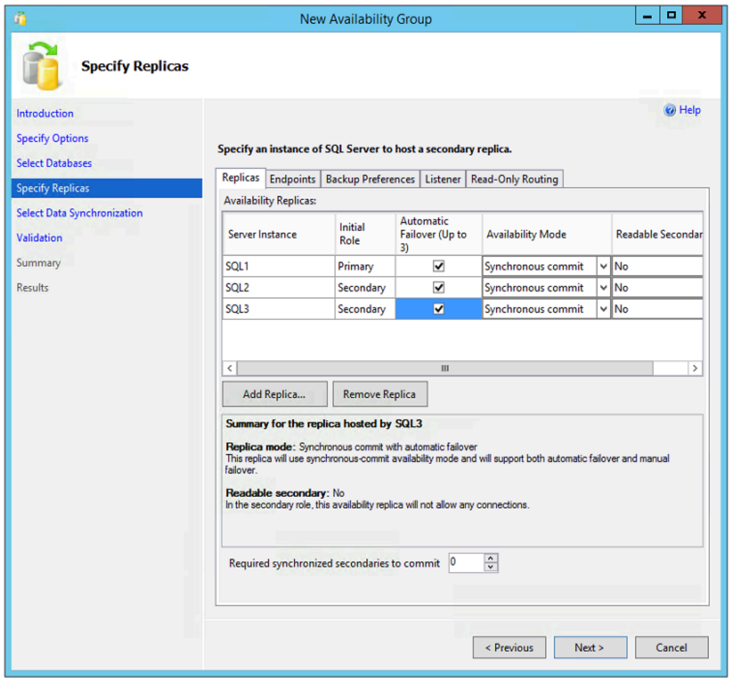
Select user databases for the availability group.

User databases on this instance of SQL Server:

	Name	Size	Status	Password
✓	AlwaysOn_DB	16.0 MB	Meets prerequisites	

5. On the **Specify Replicas** page, add the other two SQL Server instances by clicking the **Add Replica** button and entering SQL2 and SQL3.

6. Select the **Automatic Failover** check box for all of the instances.



Specify an instance of SQL Server to host a secondary replica.

Availability Replicas:

Server Instance	Initial Role	Automatic Failover (Up to 3)	Availability Mode	Readable Secondar
SQL1	Primary	✓	Synchronous commit	No
SQL2	Secondary	✓	Synchronous commit	No
SQL3	Secondary	✓	Synchronous commit	No

Summary for the replica hosted by SQL3

Replica mode: Synchronous commit with automatic failover
This replica will use synchronous-commit availability mode and will support both automatic failover and manual failover.

Readable secondary: No
In the secondary role, this availability replica will not allow any connections.

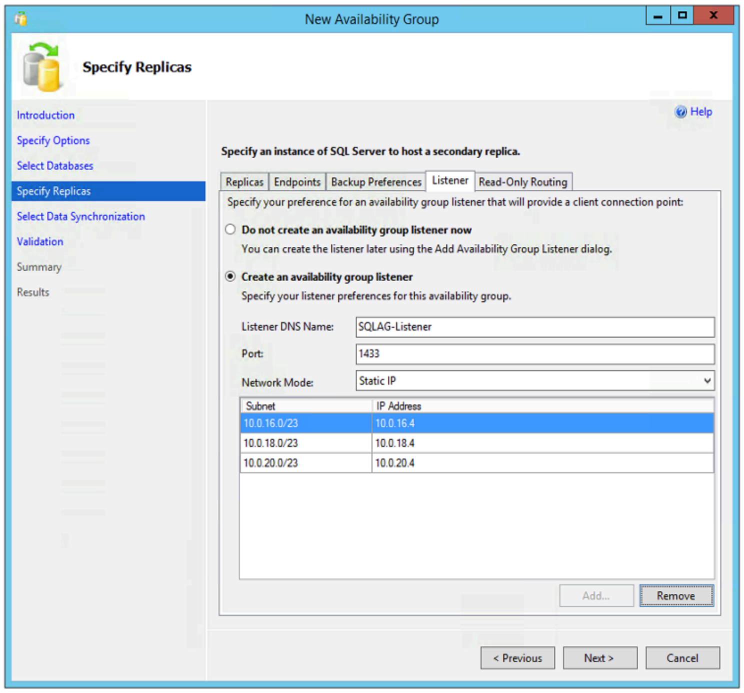
Required synchronized secondaries to commit 0

7. Click the **Listener** tab, select the **Create an availability group listener** option, and specify the following values:

   A. Specify the name of the listener, for example, `SQLAG-Listener`.

   B. Specify the port (port 1433 by default).

C. Add the private IP addresses assigned to the nodes for the listener (10.0.16.4, 10.0.18.4, and 10.0 20.4 in this example). Ensure that the IP addresses you use match those created either by the Terraform templates or by you manually.



Specify an instance of SQL Server to host a secondary replica.

Specify your preference for an availability group listener that will provide a client connection point:

( ) Do not create an availability group listener now
You can create the listener later using the Add Availability Group Listener dialog.

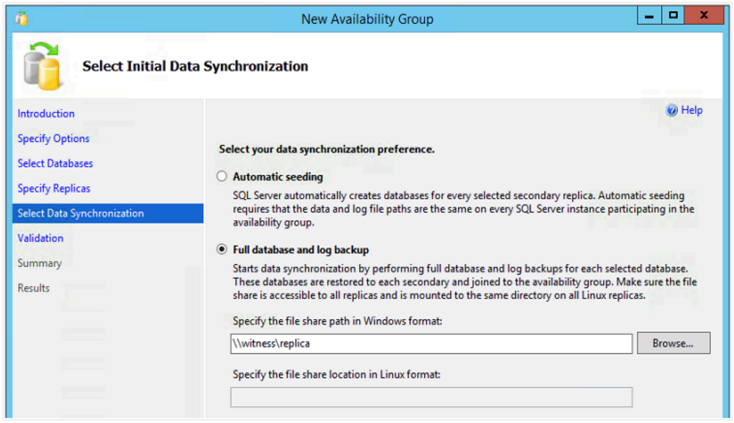
(●) Create an availability group listener
Specify your listener preferences for this availability group.

Listener DNS Name: SQLAG-Listener
Port: 1433
Network Mode: Static IP

Subnet	IP Address
10.0.16.0/23	10.0.16.4
10.0.18.0/23	10.0.18.4
10.0.20.0/23	10.0.20.4

8. Click **Next**.

9. On the **Select Initial Data Synchronization** page, select the **Full database and log backup** option, enter `\\witness\replica` in the path text box, and then click **Next**.



Select your data synchronization preference.

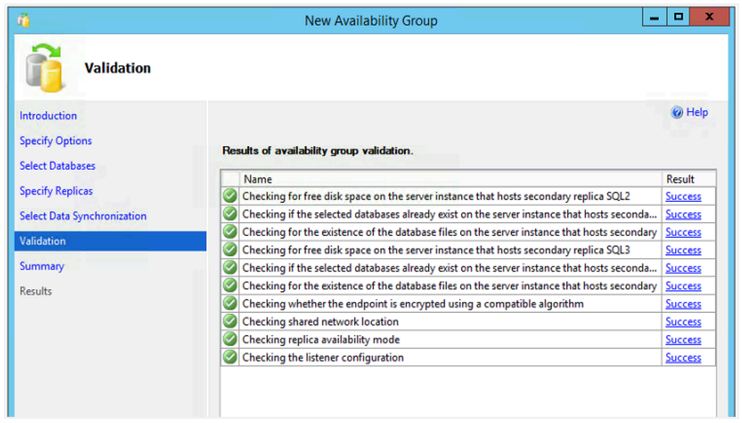
( ) Automatic seeding
SQL Server automatically creates databases for every selected secondary replica. Automatic seeding requires that the data and log file paths are the same on every SQL Server instance participating in the availability group.

(●) Full database and log backup
Starts data synchronization by performing full database and log backups for each selected database. These databases are restored to each secondary and joined to the availability group. Make sure the file share is accessible to all replicas and is mounted to the same directory on all Linux replicas.

Specify the file share path in Windows format:
\\witness\replica

Specify the file share location in Linux format:

10. On the **Validation** page, ensure that all the tests show as passed with success, and then click **Next**.



Results of availability group validation.

	Name	Result
✓	Checking for free disk space on the server instance that hosts secondary replica SQL2	Success
✓	Checking if the selected databases already exist on the server instance that hosts seconda...	Success
✓	Checking for the existence of the database files on the server instance that hosts secondary	Success
✓	Checking for free disk space on the server instance that hosts secondary replica SQL3	Success
✓	Checking if the selected databases already exist on the server instance that hosts seconda...	Success
✓	Checking for the existence of the database files on the server instance that hosts secondary	Success
✓	Checking whether the endpoint is encrypted using a compatible algorithm	Success
✓	Checking shared network location	Success
✓	Checking replica availability mode	Success
✓	Checking the listener configuration	Success

11. On the **Summary** page, review the actions to be performed and click **Finish**.

    If the environment has been set up successfully, all the actions complete successfully.

12. Click **Close**.

Change the HostRecordTTL Setting

By default, clients cache the Cluster DNS records for 20 minutes. This means that if the active node in your availability group goes offline, it can take that long for the clients to receive the IP address of the newly promoted active node. You can reduce this time by changing the Host Record Time To Live (HostRecordTTL) setting. The following example changes it to 5 minutes. You can specify a lower amount, but that might result in increased traffic to the domain name servers.

1. Use the following PowerShell command to change the HostRecordTTL setting to 300 seconds:

```
Get-ClusterResource SQLAG_AGListener | Set-ClusterParameter HostRecordTTL
300
```

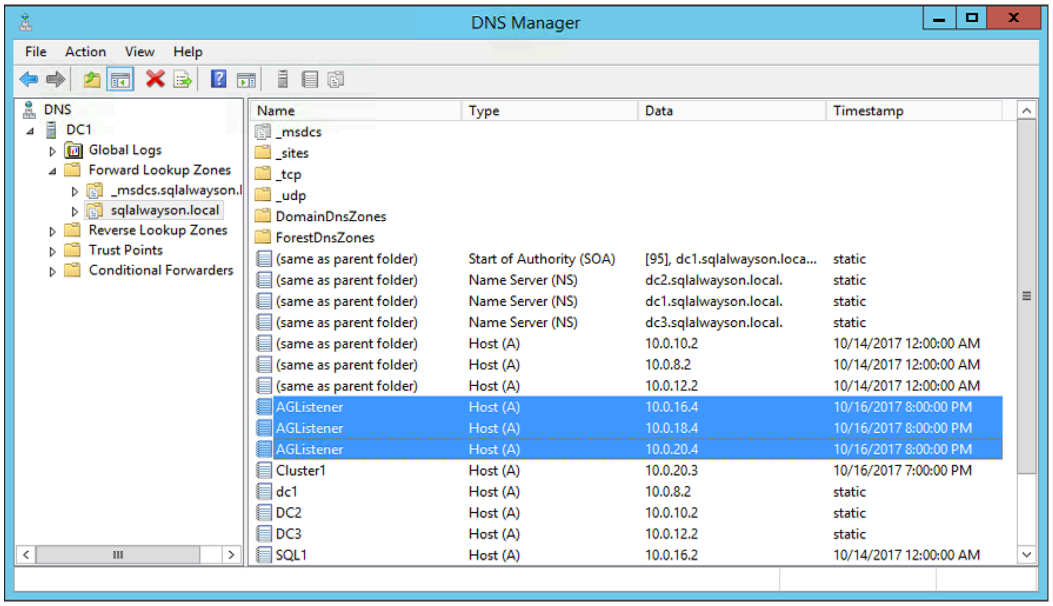
2. Verify that the setting has been recorded:

```
Get-ClusterResource SQLAG_AGListener | Get-ClusterParameter
```

3. Restart the listener for the new parameter to take effect:

```
Stop-ClusterResource SQLAG_AGListener | Start-ClusterResource
SQLAG_AGListener
```

4. Verify that the newly created listener has been registered properly with DNS by running the DNS Manager tool on one of the domain controllers.



Name	Type	Data	Timestamp
_msdcs			
_sites			
_tcp			
_udp			
DomainDnsZones			
ForestDnsZones			
(same as parent folder)	Start of Authority (SOA)	[95], dc1.sqlalwayson.loca...	static
(same as parent folder)	Name Server (NS)	dc2.sqlalwayson.local.	static
(same as parent folder)	Name Server (NS)	dc1.sqlalwayson.local.	static
(same as parent folder)	Name Server (NS)	dc3.sqlalwayson.local.	static
(same as parent folder)	Host (A)	10.0.10.2	10/14/2017 12:00:00 AM
(same as parent folder)	Host (A)	10.0.8.2	10/14/2017 12:00:00 AM
(same as parent folder)	Host (A)	10.0.12.2	10/14/2017 12:00:00 AM
AGListener	Host (A)	10.0.16.4	10/16/2017 8:00:00 PM
AGListener	Host (A)	10.0.18.4	10/16/2017 8:00:00 PM
AGListener	Host (A)	10.0.20.4	10/16/2017 8:00:00 PM
Cluster1	Host (A)	10.0.20.3	10/16/2017 7:00:00 PM
dc1	Host (A)	10.0.8.2	static
DC2	Host (A)	10.0.10.2	static
DC3	Host (A)	10.0.12.2	static
SQL1	Host (A)	10.0.16.2	10/14/2017 12:00:00 AM

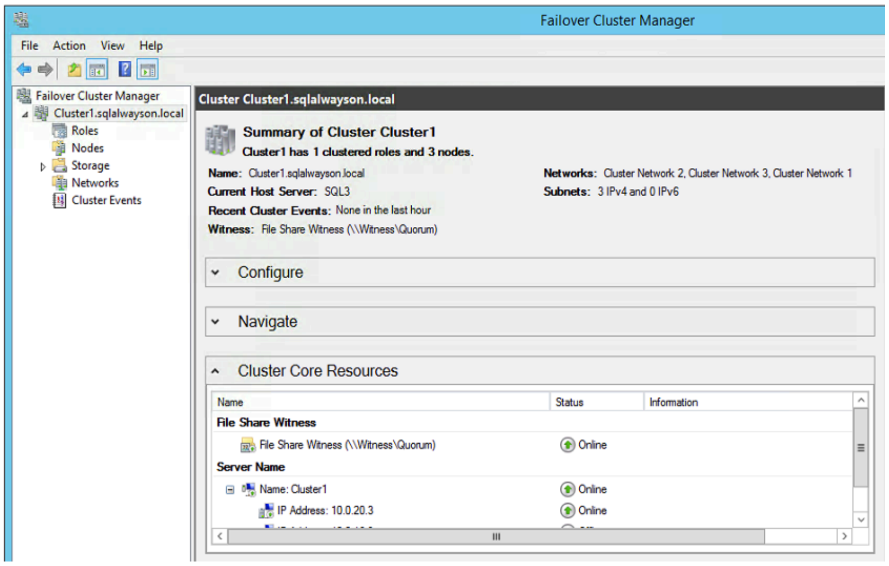
Step 7: Test the WSFC and SQL Server Always On Availability Group

Now that everything has been installed, ensure that everything is configured correctly and the cluster/availability group is performing as expected. To do that, test it in both a planned automatic failover and an unplanned disaster event.

1. Connect to one of the cluster instances (for example, SQL1).

2. Open Failover Cluster Manager, and click the cluster in the left pane.

3. Under **Cluster Core Resources**, verify that the **File Share Witness** is online.



Cluster Cluster1.sqlalwayson.local

Summary of Cluster Cluster1

Cluster1 has 1 clustered roles and 3 nodes.

Name: Cluster1.sqlalwayson.local

Current Host Server: SQL3

Recent Cluster Events: None in the last hour

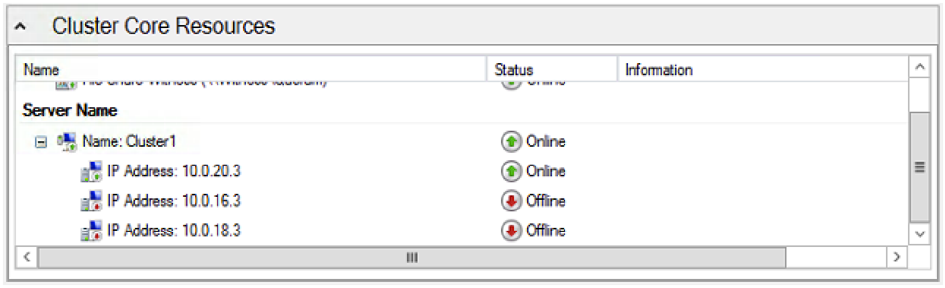
Witness: File Share Witness (\\Witness\Quorum)

Networks: Cluster Network 2, Cluster Network 3, Cluster Network 1

Subnets: 3 IPv4 and 0 IPv6

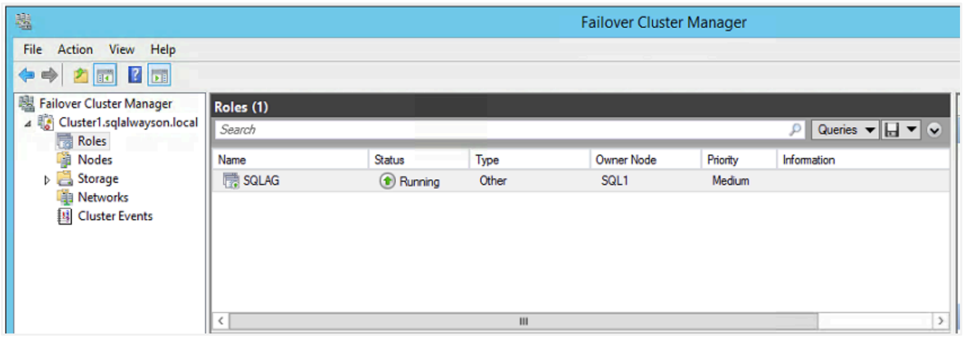
Name	Status	Information
File Share Witness		
File Share Witness (\\Witness\Quorum)	Online	
Server Name		
Name: Cluster1	Online	
IP Address: 10.0.20.3	Online	

4. In the same section, verify that one of the three IP addresses is online, and that two are offline. The IP address that is online (10.0.20.3 in the following screenshot) is the active instance (SQL3 in this example).



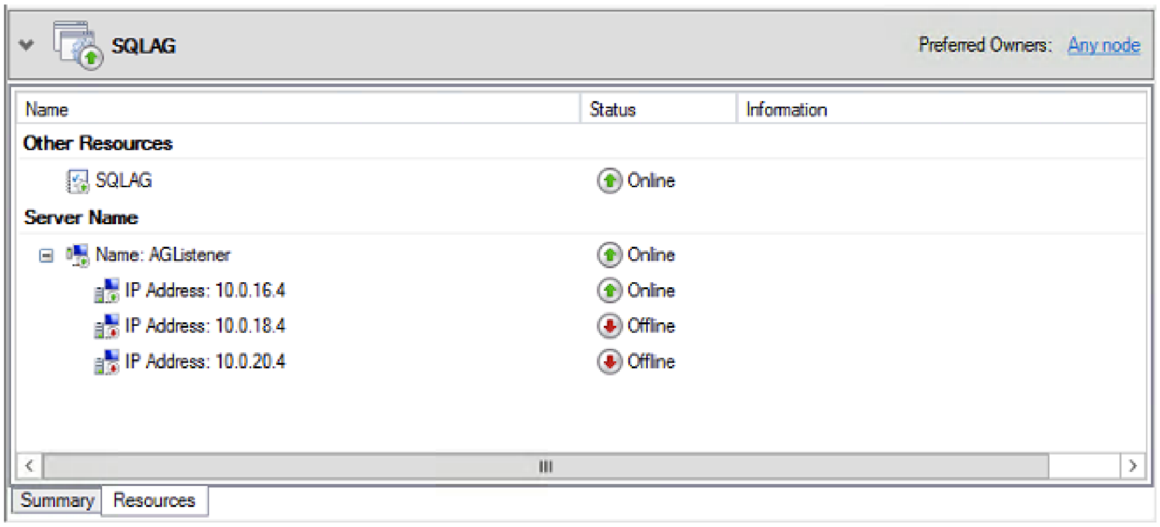
Name	Status	Information
Server Name		
Name: Cluster1	Online	
IP Address: 10.0.20.3	Online	
IP Address: 10.0.16.3	Offline	
IP Address: 10.0.18.3	Offline	

5. To verify that the availability group listener is running, click **Roles** in the left pane of Failover Cluster Manager. The status of the SQLAG role should show **Running**. If it doesn't, right-click it and select **Start Role**.



Name	Status	Type	Owner Node	Priority	Information
SQLAG	Running	Other	SQL1	Medium	

6. At the bottom of the **SQLAG** section, click the **Resources** tab, and verify that one of the IP addresses is online, while the other two are offline. Again, this is normal and shows which instance is active (SQL 1 in this example).



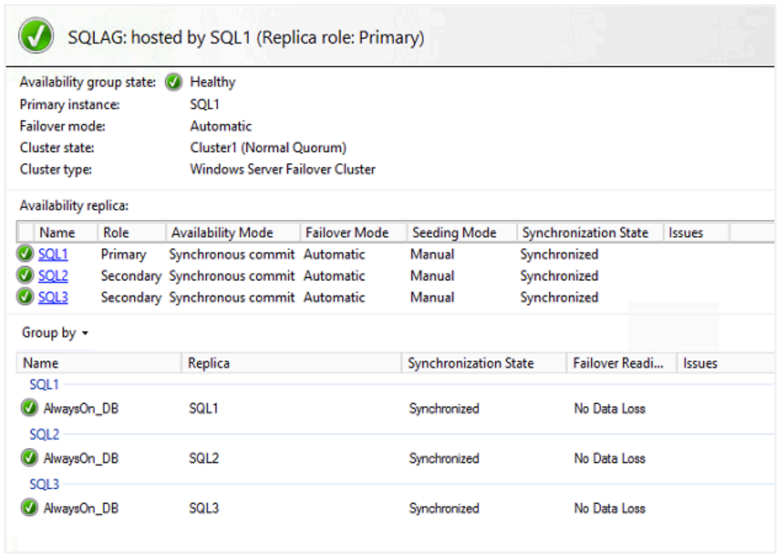
SQLAG — Preferred Owners: Any node

Name	Status	Information
Other Resources		
SQLAG	Online	
Server Name		
Name: AGListener	Online	
IP Address: 10.0.16.4	Online	
IP Address: 10.0.18.4	Offline	
IP Address: 10.0.20.4	Offline	

7. Open SQL Server Management Studio and connect to the active Always On Listener instance.

8. In the Object Explorer, expand **AlwaysOn High Availability** and ensure that SQLAG is marked as **Primary**. (If it is marked as secondary, connect to one of the other instances). Right-click **SQLAG (Primary)** and select **Show dashboard**.

9. In the dashboard, verify that the synchronization state of all the replicas is **Synchronized**.



SQLAG: hosted by SQL1 (Replica role: Primary)

Availability group state:	Healthy
Primary instance:	SQL1
Failover mode:	Automatic
Cluster state:	Cluster1 (Normal Quorum)
Cluster type:	Windows Server Failover Cluster

Availability replica:

Name	Role	Availability Mode	Failover Mode	Seeding Mode	Synchronization State	Issues
SQL1	Primary	Synchronous commit	Automatic	Manual	Synchronized	
SQL2	Secondary	Synchronous commit	Automatic	Manual	Synchronized	
SQL3	Secondary	Synchronous commit	Automatic	Manual	Synchronized	

Group by

Name	Replica	Synchronization State	Failover Readi...	Issues
SQL1				
AlwaysOn_DB	SQL1	Synchronized	No Data Loss	
SQL2				
AlwaysOn_DB	SQL2	Synchronized	No Data Loss	
SQL3				
AlwaysOn_DB	SQL3	Synchronized	No Data Loss	

10. Verify that all the resources in the cluster are on the SQL1 instance by running the following PowerShell command:

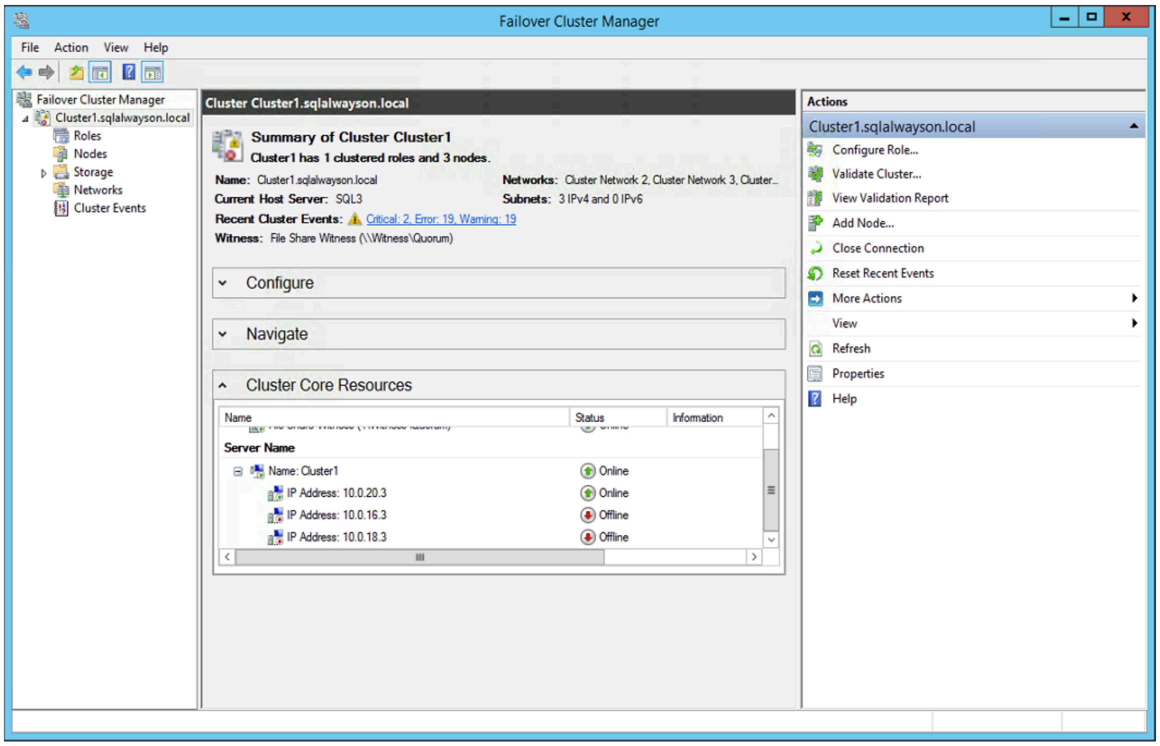
```
Get-ClusterGroup | Move-ClusterGroup SQL1
```

11. Connect to the SQL2 instance and launch Failover Cluster Manager.

12. Verify that SQL1 is still the active instance on both the cluster IP address and the SQLAG role.

13. From the Oracle Cloud Infrastructure console, shut down the SQL1 instance.

14. In Failover Cluster Manager on SQL2, verify that one of the other instances has taken over the cluster and availability group.

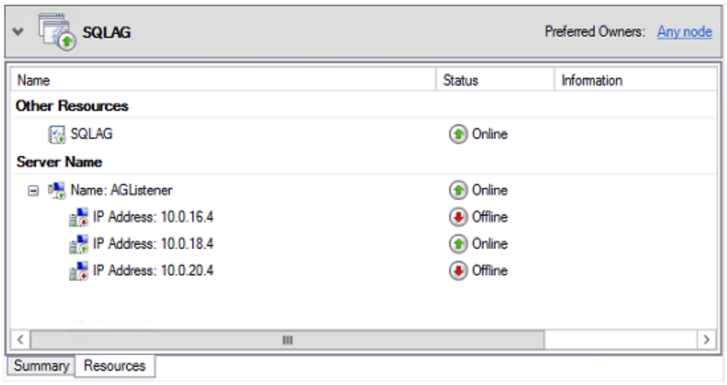


Cluster Cluster1.sqlalwayson.local

Summary of Cluster Cluster1
Cluster1 has 1 clustered roles and 3 nodes.

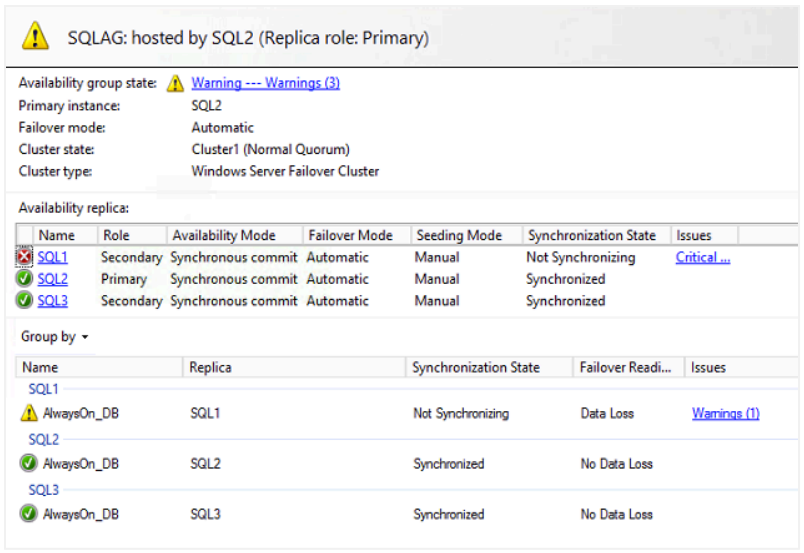
Name: Cluster1.sqlalwayson.local
Current Host Server: SQL3
Recent Cluster Events: Critical: 2, Error: 19, Warning: 19
Witness: File Share Witness (\\Witness\Quorum)
Networks: Cluster Network 2, Cluster Network 3, Cluster...
Subnets: 3 IPv4 and 0 IPv6

Name	Status	Information
Server Name		
Name: Cluster1	Online	
IP Address: 10.0.20.3	Online	
IP Address: 10.0.16.3	Offline	
IP Address: 10.0.18.3	Offline	


SQLAG Preferred Owners: Any node

Name	Status	Information
Other Resources		
SQLAG	Online	
Server Name		
Name: AGListener	Online	
IP Address: 10.0.16.4	Offline	
IP Address: 10.0.18.4	Online	
IP Address: 10.0.20.4	Offline	

15. Open SQL Server Management Studio and launch the Always On dashboard. The Primary replica should now be one of the other instances and the SQL1 instance should show critical and not synchronizing.


SQLAG: hosted by SQL2 (Replica role: Primary)

Availability group state:	Warning --- Warnings (3)
Primary instance:	SQL2
Failover mode:	Automatic
Cluster state:	Cluster1 (Normal Quorum)
Cluster type:	Windows Server Failover Cluster

Availability replica:

Name	Role	Availability Mode	Failover Mode	Seeding Mode	Synchronization State	Issues
SQL1	Secondary	Synchronous commit	Automatic	Manual	Not Synchronizing	Critical ...
SQL2	Primary	Synchronous commit	Automatic	Manual	Synchronized	
SQL3	Secondary	Synchronous commit	Automatic	Manual	Synchronized	

Group by

Name	Replica	Synchronization State	Failover Readi...	Issues
SQL1				
AlwaysOn_DB	SQL1	Not Synchronizing	Data Loss	Warnings (1)
SQL2				
AlwaysOn_DB	SQL2	Synchronized	No Data Loss	
SQL3				
AlwaysOn_DB	SQL3	Synchronized	No Data Loss	

Appendix A: Using the Terraform Templates

The Terraform templates used in this paper are located at https://github.com/oracle/terraform-provider-oci/tree/master/docs/solutions/sql_server_always_on_availability_groups.

Ensure that Terraform and the Oracle Cloud Infrastructure provider are configured and working properly. Instructions for setting up Terraform and the provider are located at https://github.com/oracle/terraform-provider-oci/blob/master/README.md.

The Terraform consists of five configuration files and several modules. Most of the files should not require modification (unless you want to change the naming conventions, for example). In fact, you should need to make changes only to the `tfvars` file, which contains the configuration specific to your environment. The following table lists the variables that you need to modify and their definitions:

Variable	**Definition**	**Required?**
tenancy_id	Your tenancy OCID.	Yes
user_id	The OCID of the user that is allowed API access to your tenancy.	Yes
fingerprint	The fingerprint for the preceding user.	Yes
private_key_path	The path to the private key used with the preceding user to generate the fingerprint.	Yes
private_key_password	An optional password associated with the private key.	No
compartment_id	The OCID of the compartment in which the SQL Server Always On availability group is to be created.	This is optional. If you set this value to "", a new compartment will be created using the name defined in the `configuration.tf` file.
region	The region in which to create the environment.	Yes
home_region	The home region for your tenancy. This is used to create the compartment if one does not exist.	Yes

If you want to customize the environment, you can modify the variables in the `configuration.tf` file. In this file, you can set values such as the name of the VCN, the VCN CIDR block, the compartment name, the number of availability domains to create, and the instance shapes to use in the creation of the environment.

One of the key variables in the `configuration.tf` file is the `ad_count` variable. By default, it is set to `3`, which causes Terraform to create the environment across all three availability domains in the region. Setting this value to `2` causes Terraform to create the environment in only two of the availability domains (namely, AD1 and AD2).

Appendix B: Security Rules

Three security lists are created by the Terraform scripts:

- Administration: Admin-Seclist-iad.sl
- DMZ: DMZ-Seclist-iad.sl
- SQL: SQL-Seclist-iad.sl

The specific ingress and egress rules for each of these security lists are listed in the following tables. All rules are stateful.

ADMINISTRATION SECURITY LIST: INGRESS RULES

Source	IP Protocol	Source Port Range	Destination Port Range
10.0.16.0/21	TCP	All	53
10.0.16.0/21	UDP	All	53
10.0.16.0/21	TCP	All	389
10.0.16.0/21	UDP	All	389
10.0.16.0/21	TCP	All	636
10.0.16.0/21	TCP	All	3268-3269
10.0.16.0/21	TCP	All	88
10.0.16.0/21	UDP	All	88
10.0.16.0/21	TCP	All	135
10.0.16.0/21	UDP	All	135
10.0.16.0/21	TCP	All	137
10.0.16.0/21	UDP	All	137
10.0.16.0/21	TCP	All	139
10.0.16.0/21	UDP	All	138
10.0.16.0/21	TCP	All	445
10.0.16.0/21	UDP	All	445

Source	IP Protocol	Source Port Range	Destination Port Range
10.0.16.0/21	TCP	All	464
10.0.16.0/21	UDP	All	464
10.0.16.0/21	TCP	All	49152-65535
10.0.16.0/21	UDP	All	49152-65535
10.0.8.0/21	All	All	All
10.0.0.0/21	TCP	All	3389

ADMINISTRATION SECURITY LIST: EGRESS RULES

Destination	IP Protocol	Source Port Range	Destination Port Range
10.0.0.0/21	All	All	All
10.0.8.0/21	All	All	All
10.0.16.0/21	All	All	All

DMZ SECURITY LIST: INGRESS RULE

Destination	IP Protocol	Source Port Range	Destination Port Range
0.0.0.0/0	TCP	All	3389

DMZ SECURITY LIST: EGRESS RULE

Destination	IP Protocol	Source Port Range	Destination Port Range
0.0.0.0/0	All	All	All

SQL SECURITY LIST: INGRESS RULES

Destination	IP Protocol	Source Port Range	Destination Port Range
10.0.8.0/21	TCP	All	135
10.0.8.0/21	TCP	All	137
10.0.8.0/21	UDP	All	137
10.0.8.0/21	TCP	All	445
10.0.8.0/21	TCP	All	1433-1434

Destination	IP Protocol	Source Port Range	Destination Port Range
10.0.8.0/21	UDP	All	1434
10.0.8.0/21	TCP	All	3343
10.0.8.0/21	UDP	All	3343
10.0.8.0/21	TCP	All	5022
10.0.8.0/21	TCP	All	5985
10.0.8.0/21	TCP	All	464
10.0.8.0/21	UDP	All	464
10.0.8.0/21	TCP	All	49152-65535
10.0.8.0/21	UDP	All	49152-65535
10.0.8.0/21	TCP	All	1433
10.0.16.0/21	All	All	All

SQL SECURITY LIST: EGRESS RULES

Destination	IP Protocol	Source Port Range	Destination Port Range
10.0.0.0/21	All	All	All
10.0.8.0/21	All	All	All
10.0.16.0/21	All	All	All

Deploying Microsoft SQL Server Always On Availability Groups
September 2018
Author: Barry Shilmover
Technical Contact: Steven B. Nelson