Configure a SQL 2014 AlwaysOn Availability Group Cluster

by baldwin (/users/profile/baldwin) on Dec 04, 2014 **Advanced**

Table of Contents

Introduction

Before jumping into this tutorial you will want to have first completed the following pre-requisite tutorials:

Setting Up Windows 2012 R2 Active Directory Using the DCD (/tutorials/setting-up-windows-2012-r2-active-directory-using-the-dcd/)

Setup DCD for a Two Node Cluster (/tutorials/setup-dcd-for-two-node-cluster/)

Create a Windows 2012 R2 Failover Cluster on ProfitBricks (/tutorials/create-a-windows-2012-r2-failover-cluster-on-profitbricks/)

We will be leveraging the Active Directory domain you created in the first tutorial and the DCD cluster you brought online in the second. By the end of this tutorial you will have a SQL cluster deployed in your datacenter. We will be using *AlwasyOn Availability Groups* with non-shared storage.

SQL 2014 Edition Differences

Microsoft provides a great overview of the differences between SQL editions here (http://msdn.microsoft.com/en-us/library/cc645993.aspx). This page breaks down the features in SQL and the edition in which they are available.

For the features we're using in this tutorial you will need to use Enterprise Edition. We plan on covering how to setup a cluster using AlwaysOn Failover Cluster Instances, which is available for Enterprise, Business Intelligence, and Standard editions, in a later tutorial.

AlwaysOn Availability Groups

One of the more exciting additions to SQL in the past few versions has been the introduction of *AlwaysOn Availability Groups*. An *AlwaysOn Availability Group* cluster uses and requires Windows Server Failover Clustering (WSFC). The concept is very similar to Exchange's DAG technology in that an Availability Group encompasses and supports failover of a discrete set of databases. These databases are known as your availability databases. What this means is that in the event of a failure all databases associated with a given availability group will failover to a secondary replica.

One node in a cluster will act as a primary replica with support for one to eight secondary replicas. Connections to the cluster are managed by WSFC. Clients connect to an availability group listener which determines the correct replica to send the request.

Networking

You did most of the network setup in the tutorial on how to create the environment within the DCD. The remaining steps that you will need to do before moving forward with installing SQL and configuring the cluster are:

1. Configure the private interfaces on both SQL nodes to use the Domain Controllers as the DNS servers.
2. Clear the DNS server settings for the public interface to ensure you can resolve the AD domain without issue. Any public lookups would happen via referrals on the AD DNS servers.
3. Join the AD domain. This can be done by using the following PowerShell command:

You would run:

```
Add-Computer -DomainName YOURDOMAIN
```

This will prompt you for credentials. Use the username and password for the domain administrator. Once finished you should get a response from PowerShell that indicates the changes won't take affect until you reboot.

Let's not reboot at the moment, but continue onto Windows Update which might require a reboot depending on the type of updates that are outstanding.

Windows Update

As a best practice -- and especially with cloud server instances -- I always like to run Windows Update before starting anything major. Once we're done with the SQL installation we will re-run Windows Update to ensure we capture any additional SQL updates that may need to be installed.

Go ahead and run Windows Update at this time if you want. Reboot once all updates have been installed. If there are no updates to install go ahead and reboot to commit the domain join change.

Disk Volumes

Log onto both nodes and initialize the volumes you will be using for storing your databases and transaction logs. If you followed our tutorial on provisioning a two-node cluster in the DCD then you should have three volumes total:

- a volume for the system
- a volume for the SQL databases
- a volume for the SQL transaction logs

This would be done by bringing up the *Disk Management* interface.

Setup SQL User and Groups

First, we need to create some Managed Service Accounts. These will be used for each SQL node we add to the cluster. We create three accounts per node in the cluster and associate those service accounts to the node. This can be simplified using PowerShell like so:

```
New-ADServiceAccount -Name SQL01SVCACC -Description "SQL01 Service Account" -Path "OU=ClusterObje
cts,DC=yourdomain,DC=local"
New-ADServiceAccount -Name SQL01AGTACC -Description "SQL01 Agent Account" -Path "OU=ClusterObject
s,DC=yourdomain,DC=local"
New-ADServiceAccount -Name SQL01SBWACC -Description "SQL01 Browser Service Account" -Path "OU=Clu
sterObjects,DC=yourdomain,DC=local"
New-ADServiceAccount -Name SQL02SVCACC -Description "SQL02 Service Account" -Path "OU=ClusterObje
cts,DC=yourdomain,DC=local"
New-ADServiceAccount -Name SQL02AGTACC -Description "SQL02 Agent Account" -Path "OU=ClusterObject
s,DC=yourdomain,DC=local"
New-ADServiceAccount -Name SQL02BWACC -Description "SQL02 Browser Service Account" -Path "OU=Clus
terObjects,DC=yourdomain,DC=local"
```

You will next need to associate each account with the SQL node to which it belongs.

```
Add-ADComputerServiceAccount -Identity SQL01 -ServiceAccount SQL01SVCACC,SQL01AGTACC,SQL01SBWACC
Add-ADComputerServiceAccount -Identity SQL02 -ServiceAccount SQL02SVCACC,SQL012AGTACC,SQL02SBWACC
```

The above should be ran from a server that has the AD PowerShell tools installed. We recommend installing these tools on each of the nodes using the following command.

```
Install-WindowsFeature RSAT-AD-PowerShell
```

Now it's time to install these newly created service accounts on each node. It is **important** that you are logged into the node when running these.

```
Get-ADComputerServiceAccount -Identity SQL01 | Install-ADServiceAccount
```

From the second node:

```
Get-ADComputerServiceAccount -Identity SQL02 | Install-ADServiceAccount
```

It is also handy to create a **SqlServerAdministrators** group at this time. This will be assigned during the SQL installation. Ensure that the user you are going to use to enable AlwaysOn and configure a DB is a member of this group. This includes your domain administrator if you're using that user.

Install SQL on the First Node

Since we're installing a test cluster we will be using the evaluation version of SQL 2014. You will need to ensure you have the appropriate license for the version of SQL you will be using.

You can download the evaluation version here (http://www.microsoft.com/en-us/evalcenter/evaluate-sql-server-2014). This gives you a 180-day license of SQL 2014. This is sufficient to get a feel for AlwaysOn Availability Groups. You will also need to download the latest cumulative update here (http://support.microsoft.com/kb/2772858/en-us) and extract it to a location on disk.

Ensure that the Windows Firewall is not protecting the heartbeat or private networks.

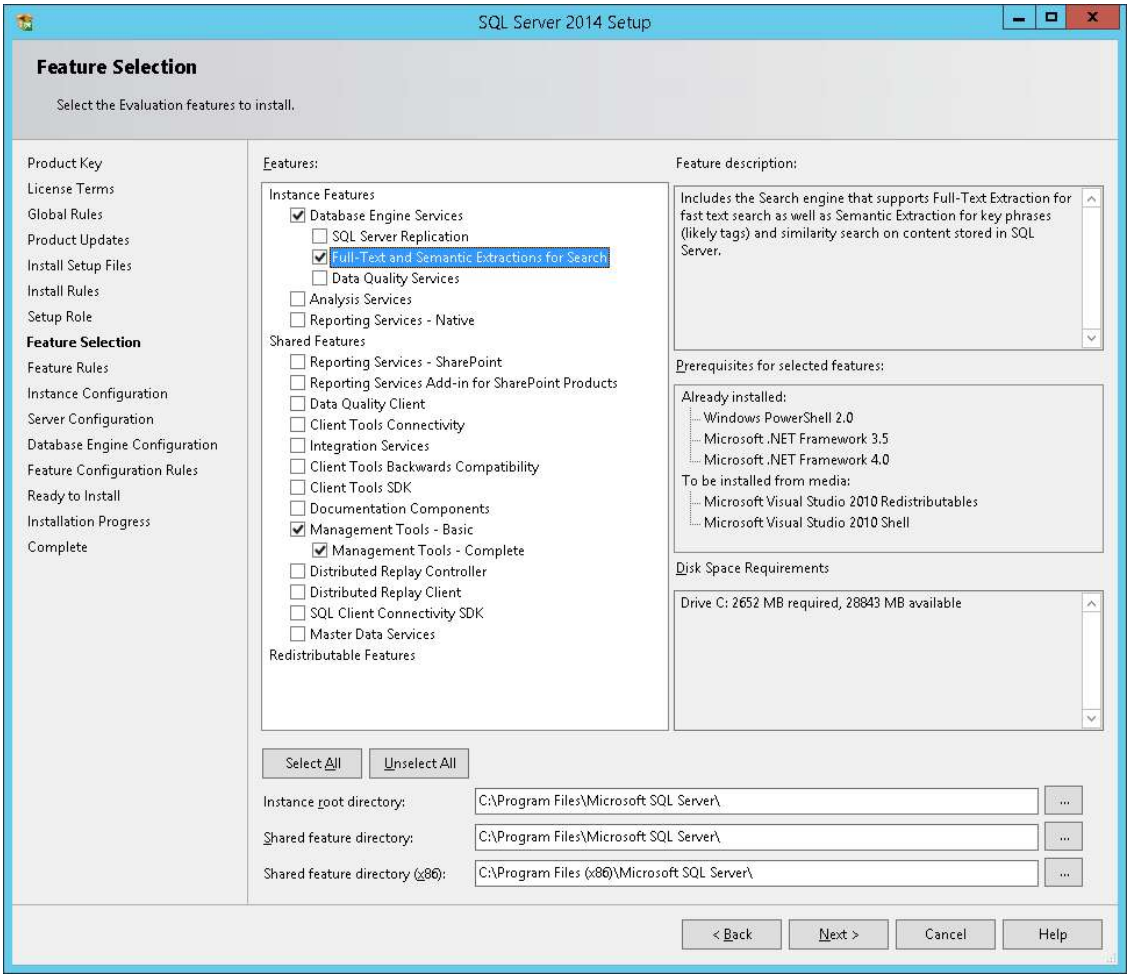
Log onto the first node and either mount or download your SQL binaries.

You will then need to run this at the command line:

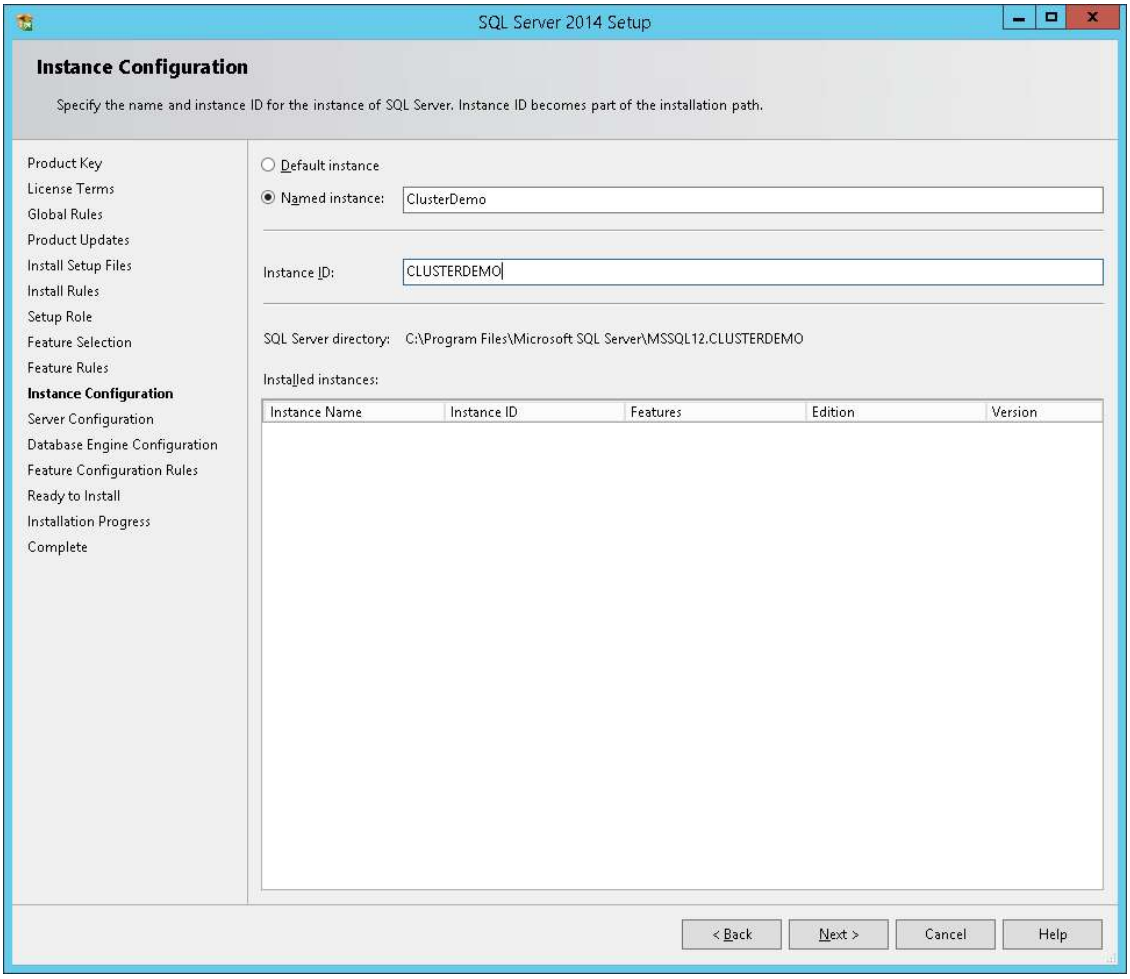
```
C:\sql\setup\file\path\setup.exe /action=install /updateenabled=true /updatesource="C:\path\to\update"
```

Once the wizard comes up:

1. Input your product key and click next.
2. Accept the terms, next.
3. Choose SQL Server Feature Installation
4. You will be installing these features:
5. Database Engine Services
6. Full-Text Search
7. Management Tools -- Complete

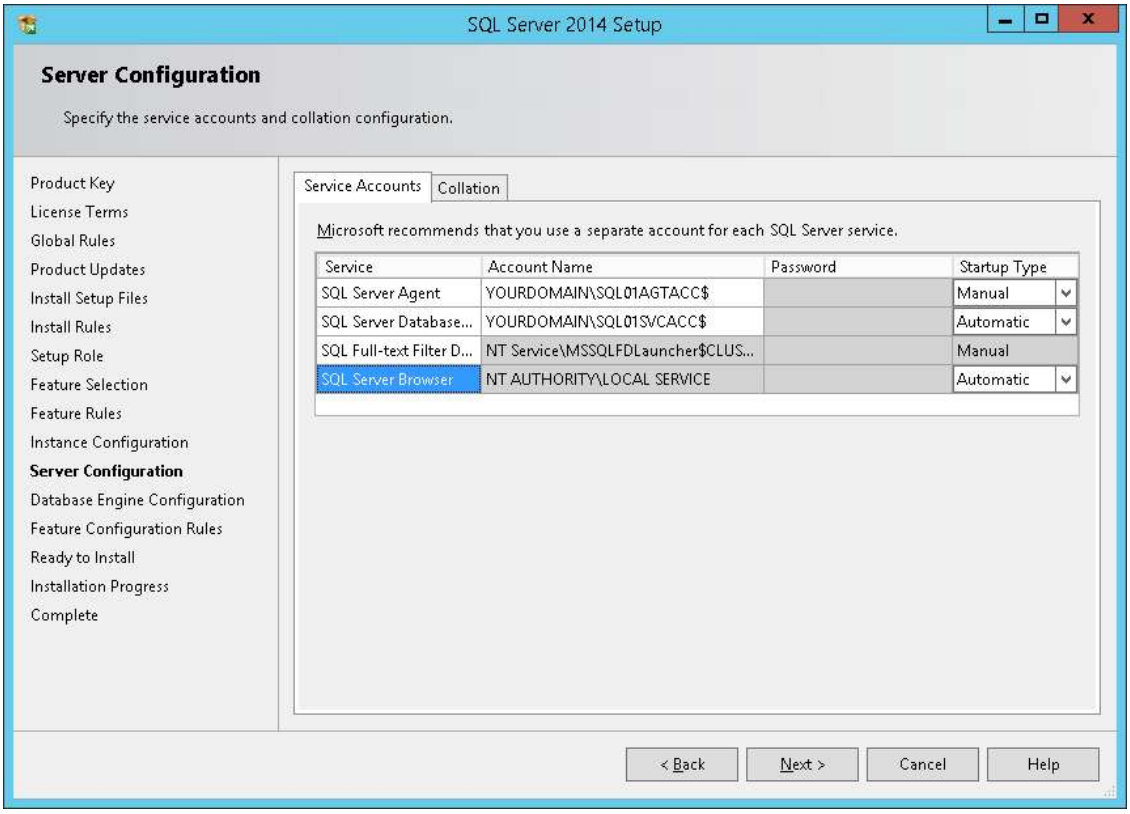


5. Choose named instance and set its name to something other than the default. You can also change your instance path at this time.

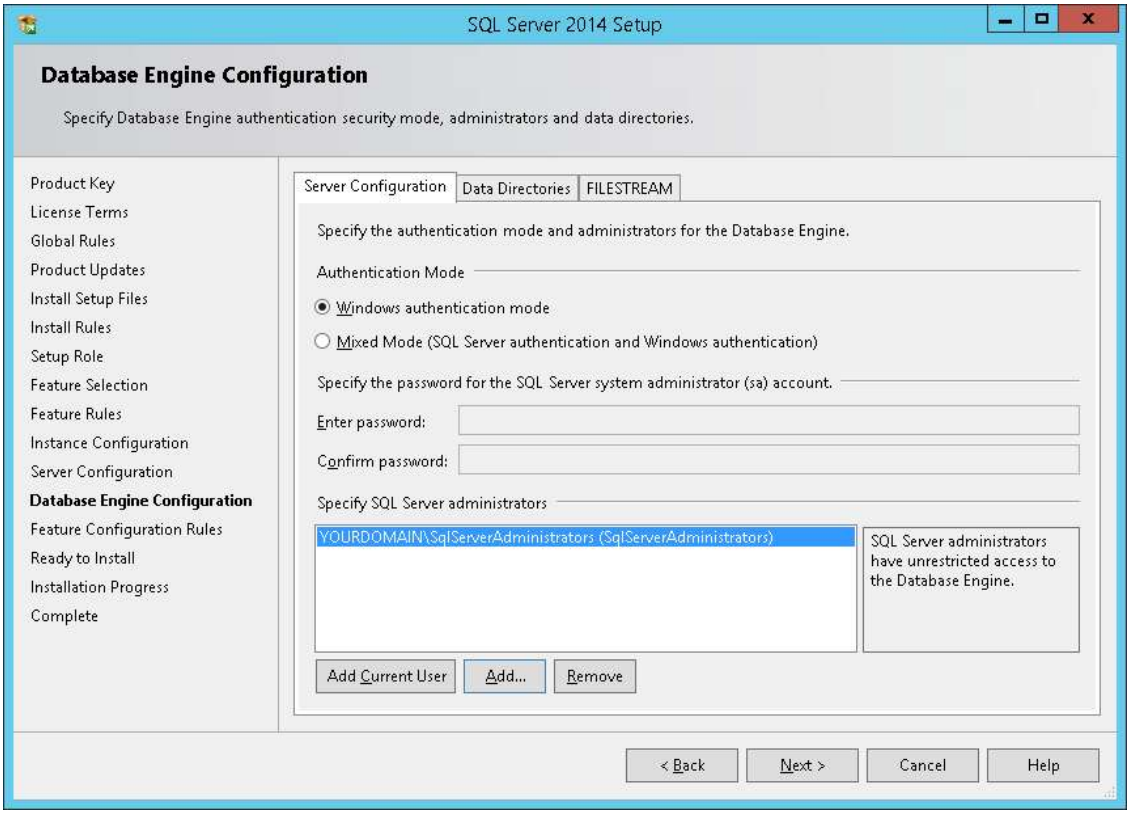


6. Set the **Account Name** for each of the service accounts.

```
| Service | Account Name | Startup Type |
| --- | --- | --- |
| SQL Server Agent | YOURDOMAIN\SQL01AgtAcc$ | Default |
| SQL Server Database Engine | YOURDOMAIN\SQL01SvcAcc$ | Default |
```

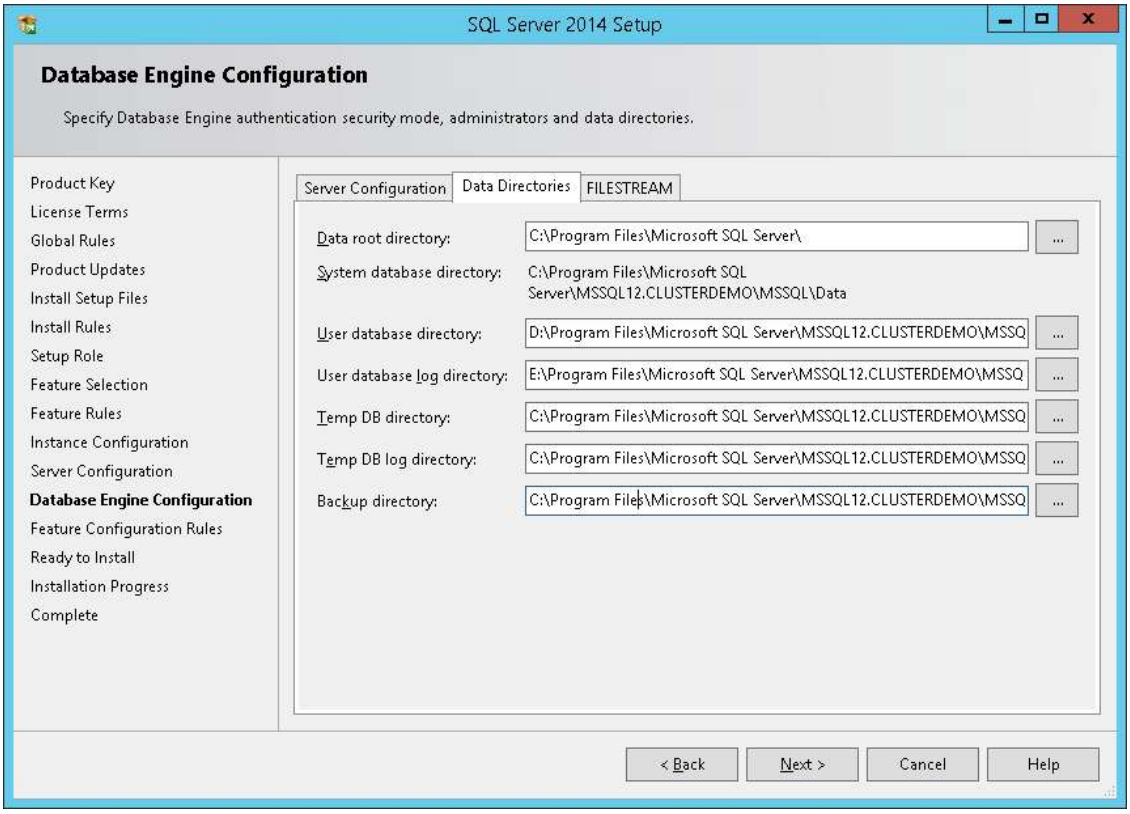


7. Choose your authentication mode. Best practices say you'd leave the default, but turning on mixed mode gives you a way into SQL when all else fails. You will also need to add the SqlServerAdministrators group as an defined SQL Server Administrator.



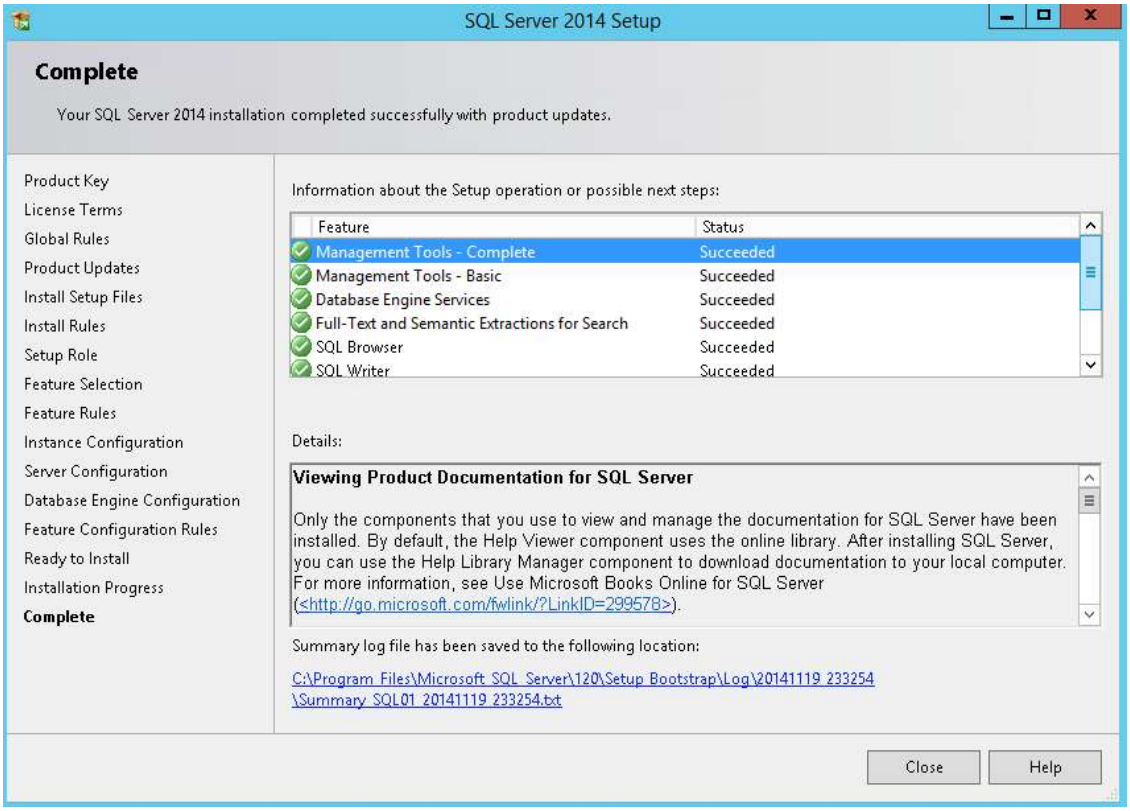
8. You will then want to update your directories so that your databases are on the second volume you've attached to your server and the transaction logs are on the third. At this time, if you choose to, you can create a fourth volume and use that as your backup location.

It is also very important to note that to use *Availability Groups* the database and transaction log paths need to remain identical. You will need to adjust the default paths to reflect the same path on both nodes.



9. Click through to complete the setup.

At this point your first node should be online.



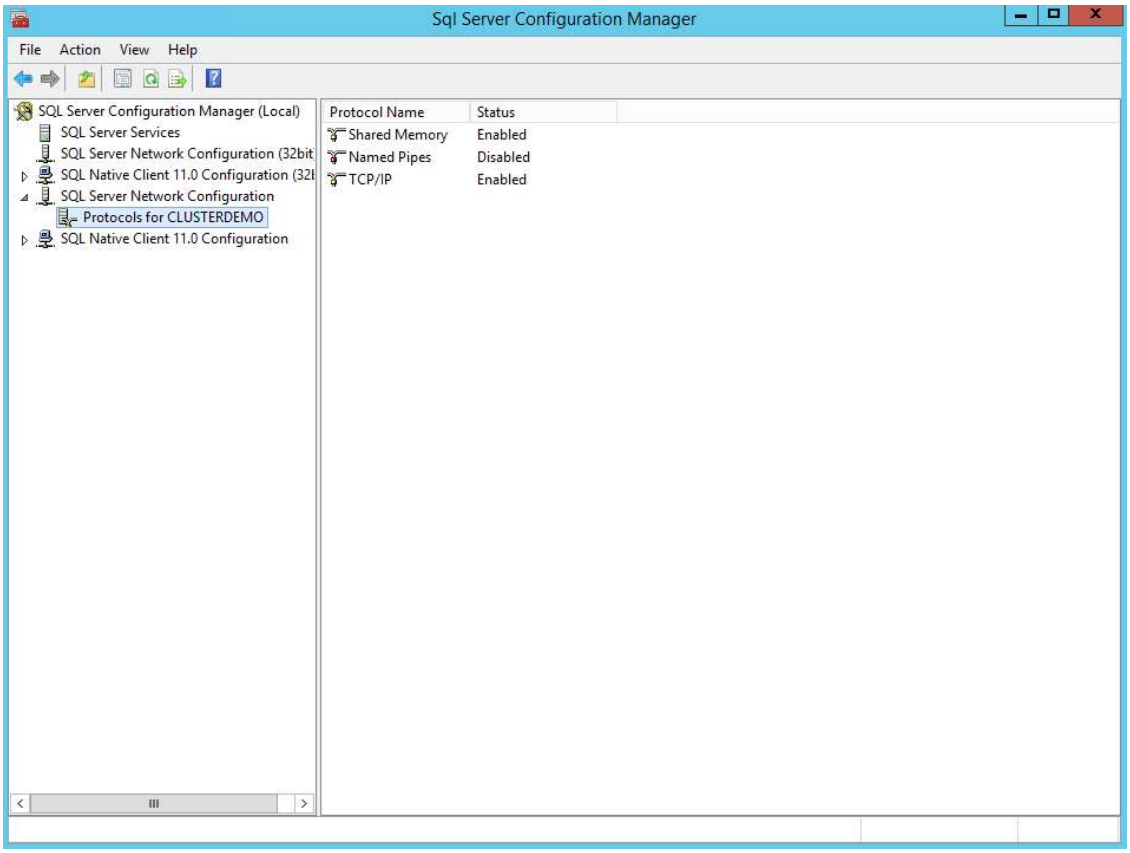
Install SQL on the Second Node

You will want to repeat the above steps on each node of your cluster. It might be handy to generate an INI or script out the installation if you're going to be doing this numerous times.

Configure SQL Networking

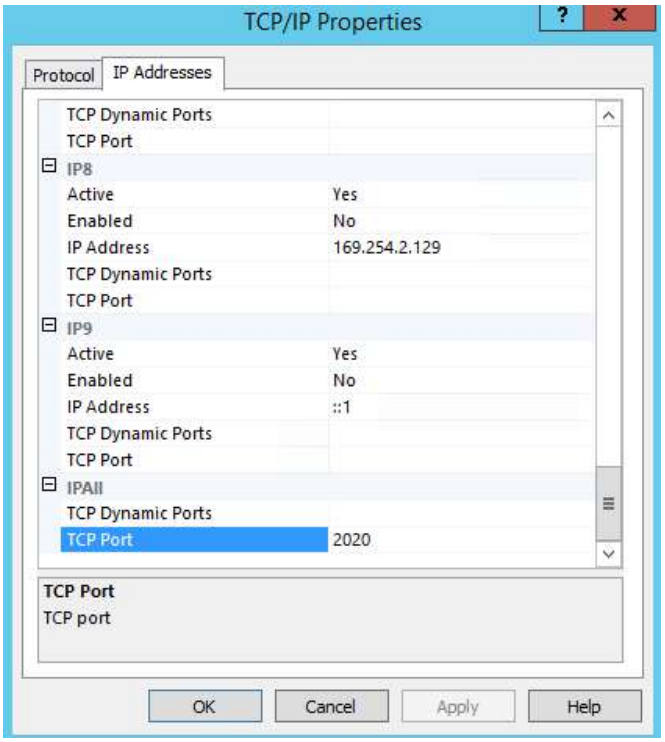
SQL needs to be configured for static ports. The following steps will need to be done on each cluster.

Bring up the Sql Server Configuration Manager.



You will want to drill down to *SQL Server Network Configuration --> Protocols for YOURINSTANCENAME*.

You will need to remove the "TCP Dynamic Ports" value 0 for all IP nodes (IP1, IP2, IP3, etc.). This can be done by simply highlighting and deleting it.



Clear the *TCP Dynamic Ports* value under IPALL.

Enter your desired static port in the *TCP Port* field. This will need to be the same port across all nodes. You can make this whatever you want. In our example we're using *2020*.

Restart SQL services.

Enabling AlwaysOn

Finally, you're ready to enable *AlwaysOn* using PowerShell:

```
Enable-SqlAlwaysOn -ServerInstance "SQLSERVER\SQLINSTANCE" -Force
```

Replace SQLSERVER with the name of your server and SQLINSTANCE with the instance named you used. Perform this step on each node. You will also need to restart your the SQL Server service.

Configure AlwaysOn for Your Database

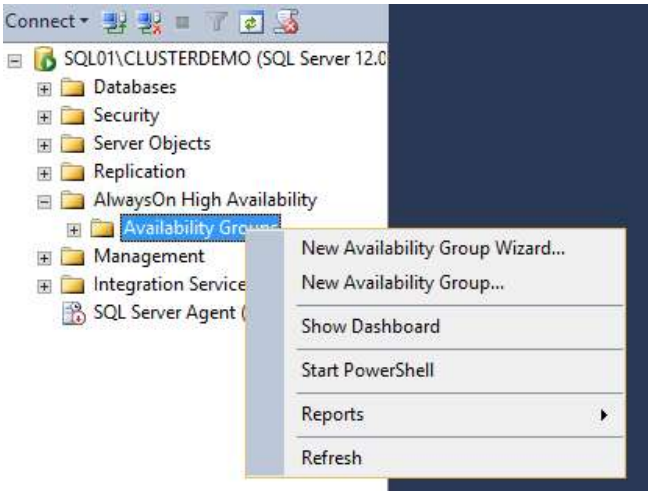
Now that your nodes are built out, networking is configured, and AlwaysOn is enabled it is time to configure a database for redundancy. This can be done from with the SQL Management Studio.

Before you start you will already need to either create a database or attach a database to the SQL instance. You will also need to perform a full backup of the database.

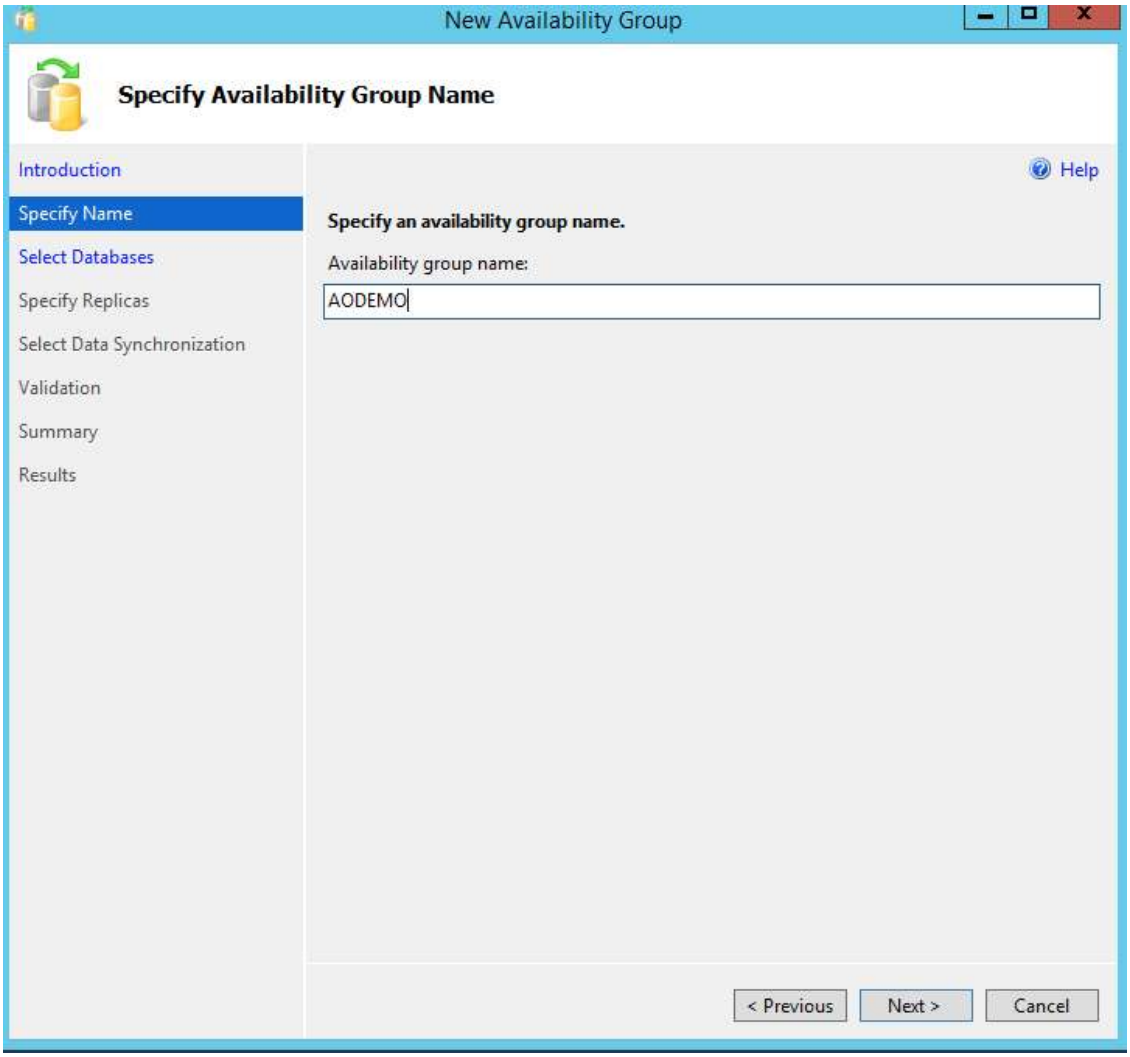
We will only create a single AlwaysOn Availability Group. We will name this availability group: AODEMO.

Log onto SQL01 and bring up the SQL Management Studio.

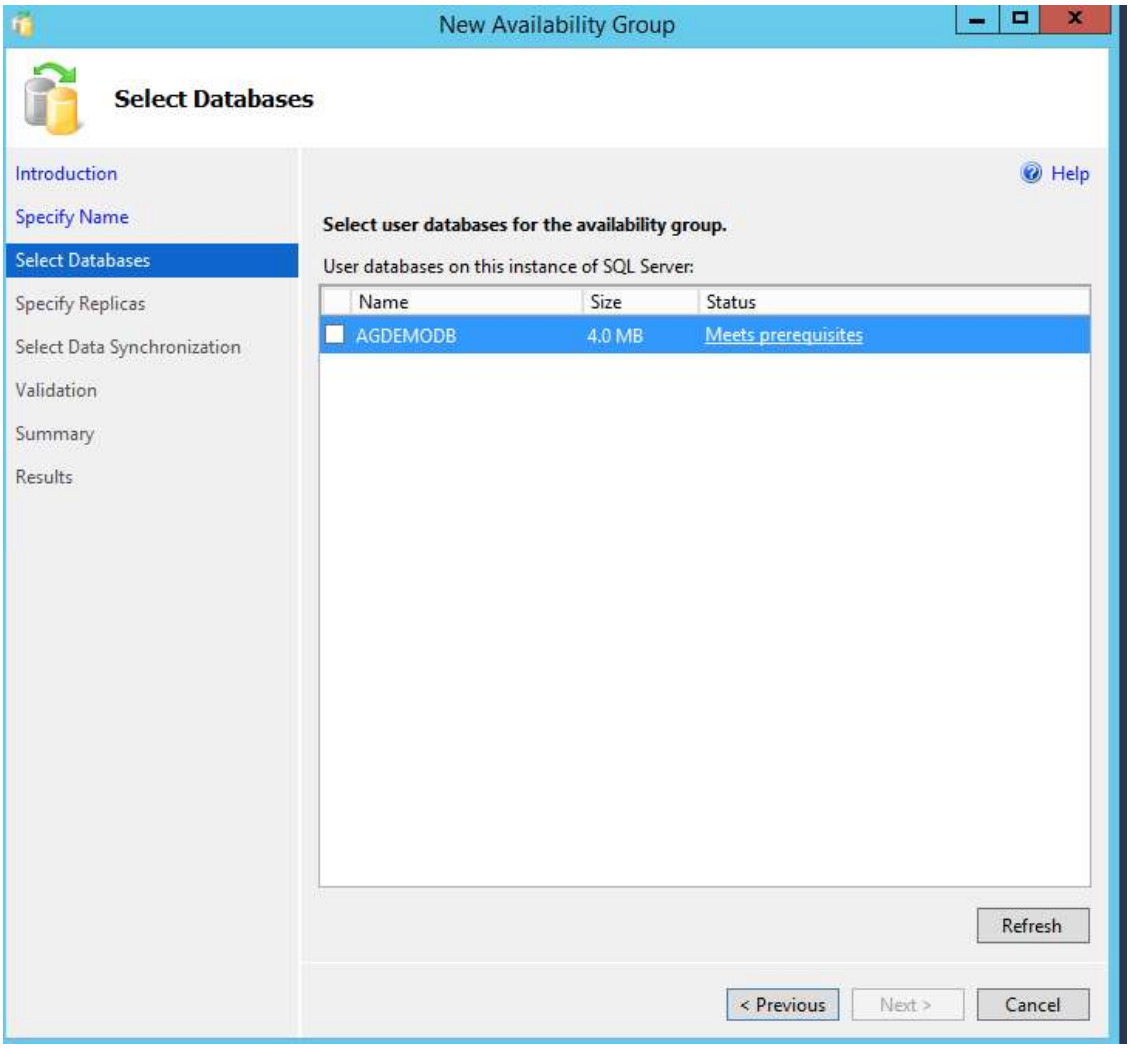
Right-click on *Availability Groups* and select *New Availability Group Wizard*.



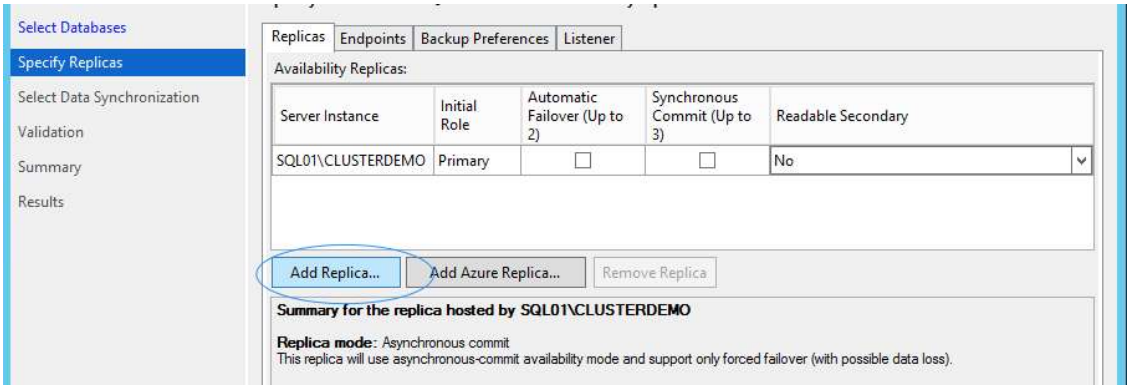
Name your AG.



Choose your database.



Click "Add Replica".

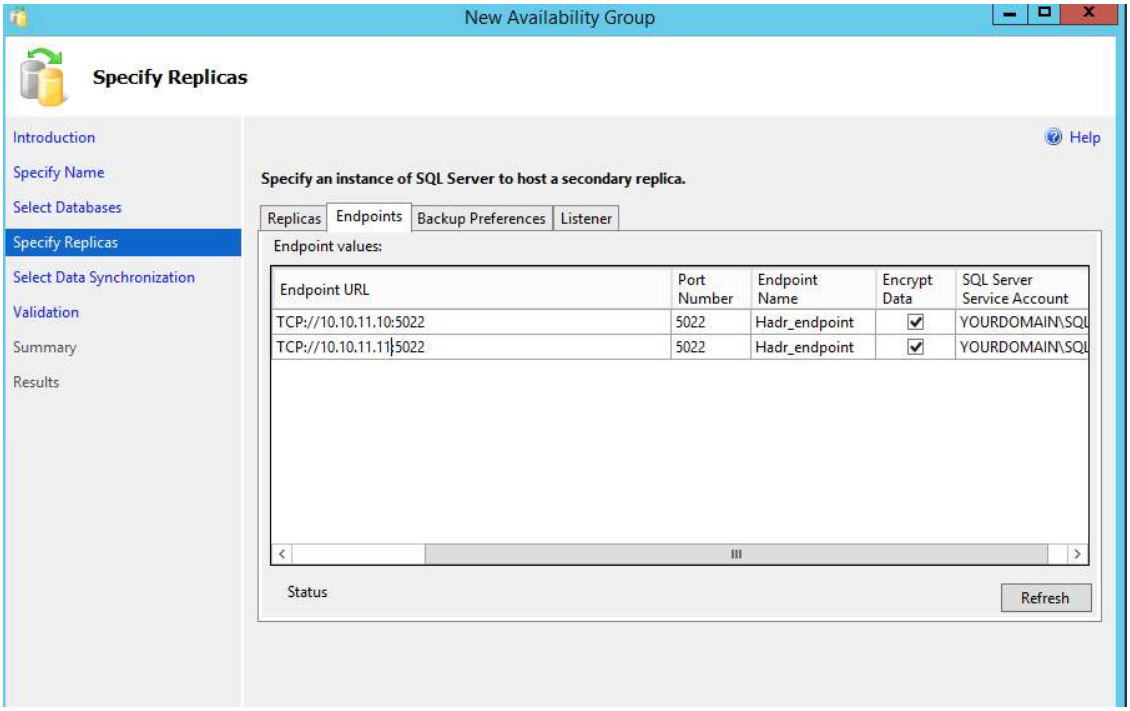


Connect to your second node. In our case this is SQL02.

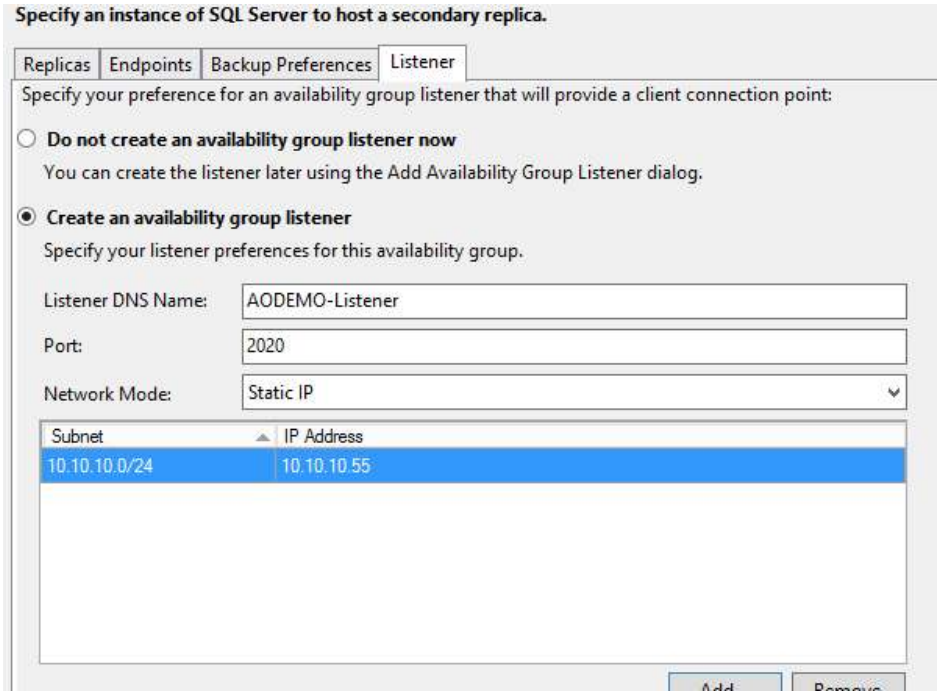
You will want to enable the following:

- Automatic Failover
- Synchronous Commit
- Readable Secondary

Click the Endpoints tab and configure your endpoints. We will be using the heartbeat network we created for node to node communication. This will serve as our replica network, too.



Click the Listener tab and configure the listener. You'll notice we're using the port we defined when configuring the networking.



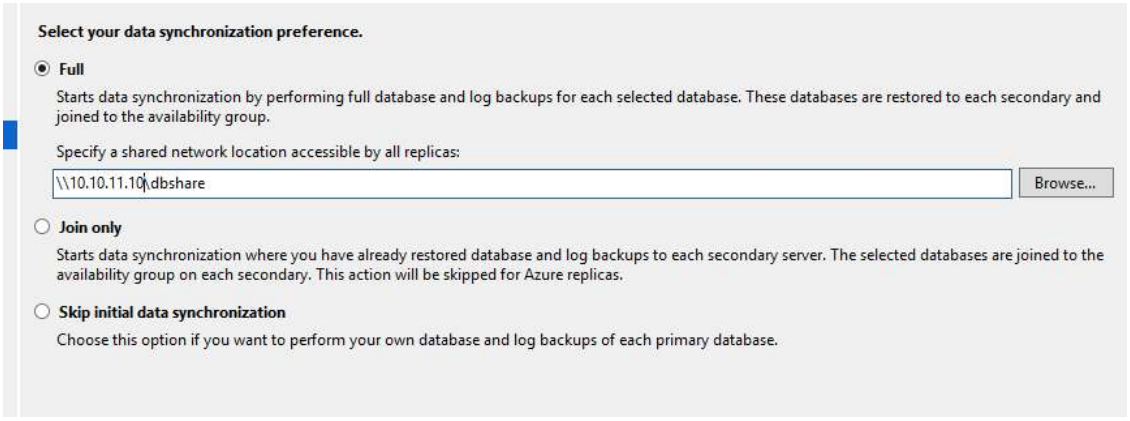
You will need to configure a data synchronization method. We recommend using **full**. You will need to ensure that your nodes are setup identically and that the database and log path are the same on all nodes. We used the file paths:

- D:\databases\
- E:\transaction_logs\

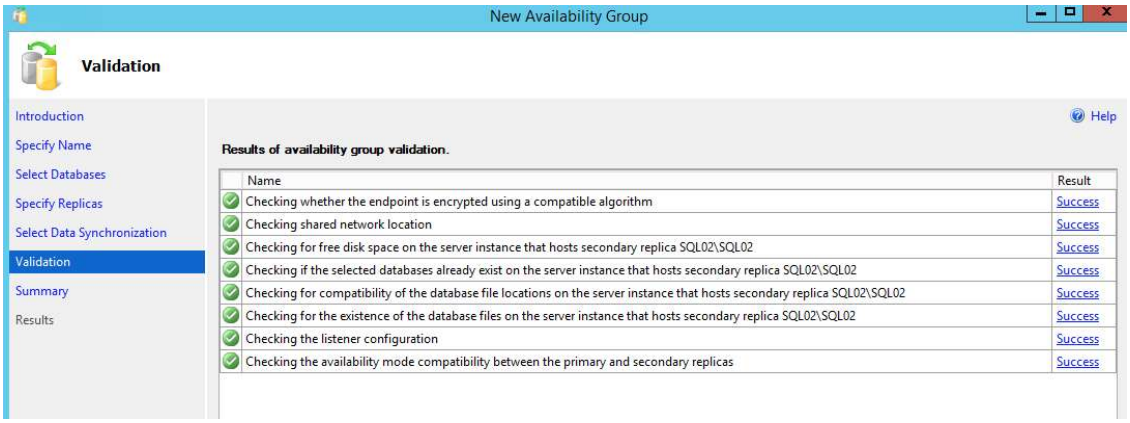
You can obviously make these paths different. The only requirement is they must be identical across all nodes in the cluster so the default paths will not work.

The default will attempt to use \SQL01 (the node you're currently on). One option is to create a DFS location for your shared content. This goes beyond the scope of this article but will be covered in the future.

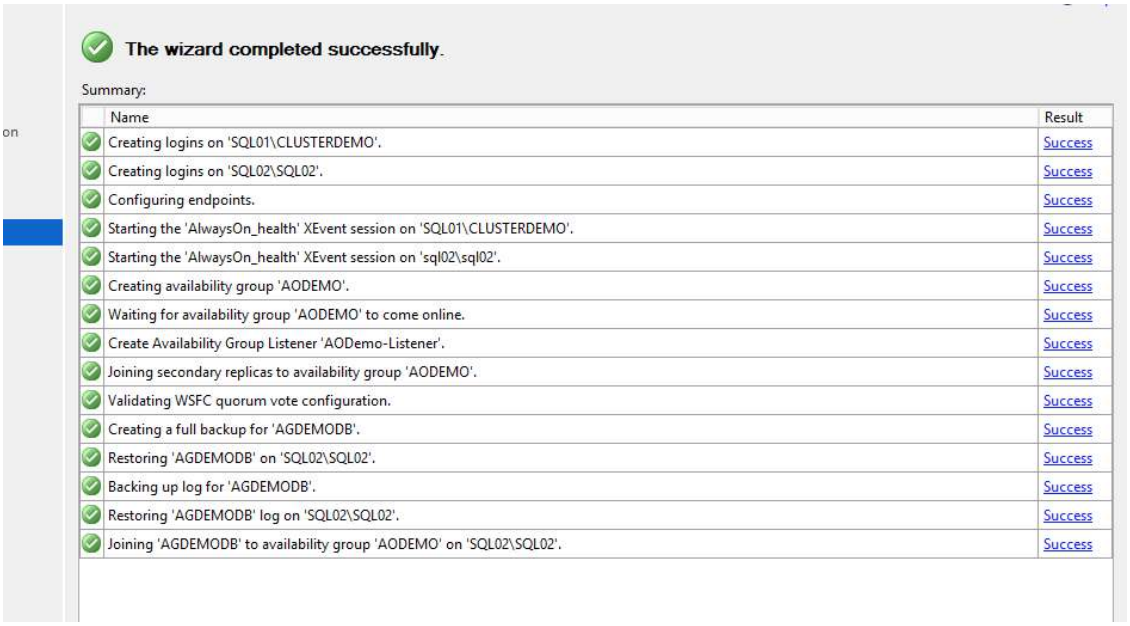
You'll notice that we're using a share available on our node to node communication network, what we called the heartbeat network.



Ensure all validation checks are green.

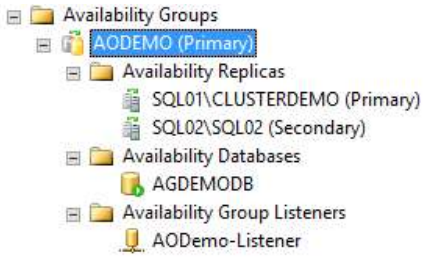


Click next and then finish. This will configure endpoints and perform various other tasks. You should see a results page like this.

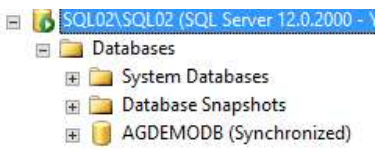


Validation

Exploring the new Availability Group in SQL Server Management Studio you should see that there's an *Availability Group* named AODEMO. Under *Availability Replicas* you should see your nodes with your first node being the primary and the other node as your secondary. There should be a single *Availability Database* named, in our case, **AGDEMODB**. Finally you should see your listener. If you were to check AD DNS this listener will be registered with the internal IP you provided.



Log onto your secondary node and bring up the SQL Server Management Studio. You should see that you have the database there as well.



Conclusion

Congratulations! At this point you should now have a database using *AlwaysOn Availability Groups*.

SHARE

☆ Subscribe (/tutorials/configure-a-sql-2014-alwayson-availability-group-cluster/subscribe/)

author:

Matt Baldwin (/users/profile/baldwin)

Dec 04, 2014
HA (/tutorials/tags/HA) , SQL (/tutorials/tags/SQL) , Windows (/tutorials/tags/Windows)

Related Tutorials

- Create a Windows 2012 R2 Failover Cluster on ProfitBricks (https://devops.profitbricks.com/tutorials/create-a-windows-2012-r2-failover-cluster-on-profitbricks/)
- Configure MongoDB Replica Set (https://devops.profitbricks.com/tutorials/configure-mongodb-replica-set/)
- Install Crate.IO on Debian 8 (https://devops.profitbricks.com/tutorials/install-crateio-on-debian-8/)
- Get Started with Crate.IO on CentOS 7 (https://devops.profitbricks.com/tutorials/get-started-with-crateio-on-centos-7/)
- Setup DCD for Two Node Cluster (https://devops.profitbricks.com/tutorials/setup-dcd-for-two-node-cluster/)

Popular Tutorials

- Install MySQL on CentOS 7 (https://devops.profitbricks.com/tutorials/install-mysql-on-centos-7/)
- Use SSH Keys with PuTTY on Windows (https://devops.profitbricks.com/tutorials/use-ssh-keys-with-putty-on-windows/)
- Install Python 3 on CentOS 7 (https://devops.profitbricks.com/tutorials/install-python-3-on-centos-7/)
- Install and Configure mod_rewrite for Apache on CentOS 7 (https://devops.profitbricks.com/tutorials/install-and-configure-mod_rewrite-for-apache-on-centos-7/)
- Getting Started with a Multi-node Kubernetes Cluster on Ubuntu (https://devops.profitbricks.com/tutorials/getting-started-with-a-multi-node-kubernetes-cluster-on-ubuntu/)
-

lanhleo147 (/users/profile/lanhleo147/) on Jan 13, 2017 commented,
Hi, I have 03 servers in there, 02 servers installed SQL Database (SQLDB1 and SQLDB2) and 01 server installed witness. - SQLBD1 is node 1 - SQLDB2 is node 2

I have question: 1- if 01 server SQLDB1 and server witness are failed. So, does database mirroring continues functioning without interruption, except that automatic failover is not possible?

2- if it can work. How can i configure in this case?

Thanks

Log In, Add a Comment (/users/login/?next=/tutorials/configure-a-sql-2014-alwayson-availability-grou