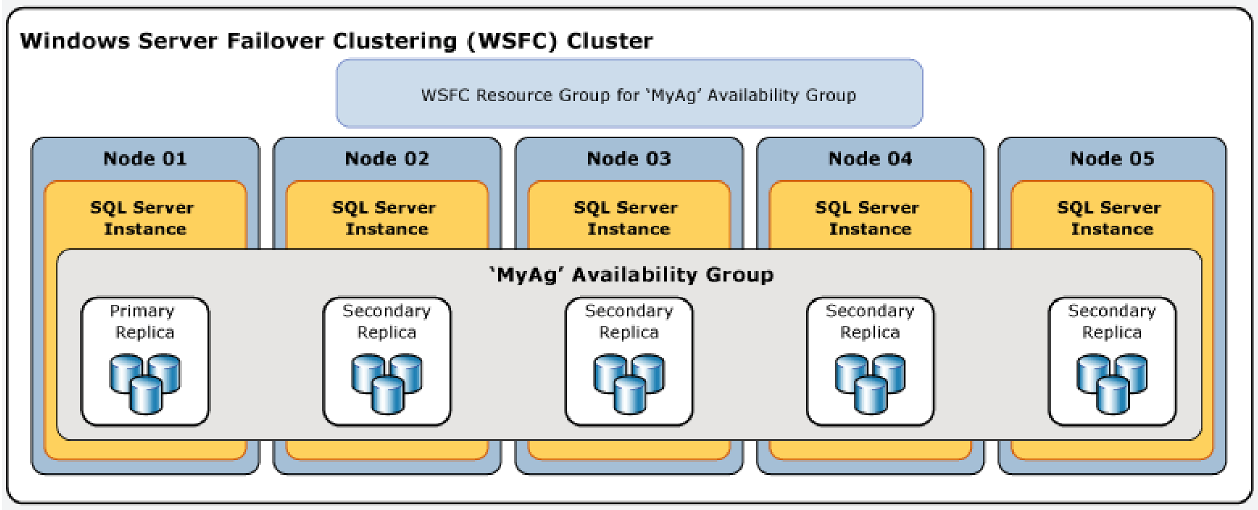


Prerequisites

Before implementing your AlwaysOn Availability Group (AG), make sure you have everything in your environment ready to go. There are several prerequisites that need to be addressed to ensure a successful deployment.

Windows

- Do not install AlwaysOn on a domain controller
- The operating system must be Windows 2012 or later
- Install all available Windows hotfixes on every server (replica)
- Windows Server Failover Cluster (WSFC) must be installed on every replica

SQL Server

- Each server (replica) must be a node in the WSFC
- No replica can run Active Directory services
- Each replica must run on comparable hardware that can handle identical workloads
- Each instance must run the same version of SQL Server, and have the same SQL Server collation
- The account that runs SQL Services should be a domain account

Network

- It is recommended to use the same network links for communication between WSFC nodes and AlwaysOn replicas

Databases in the AG

- user databases (no system databases)
- read/write
- multi-user
- AUTO_CLOSE disabled
- full recovery mode
- not configured for database mirroring

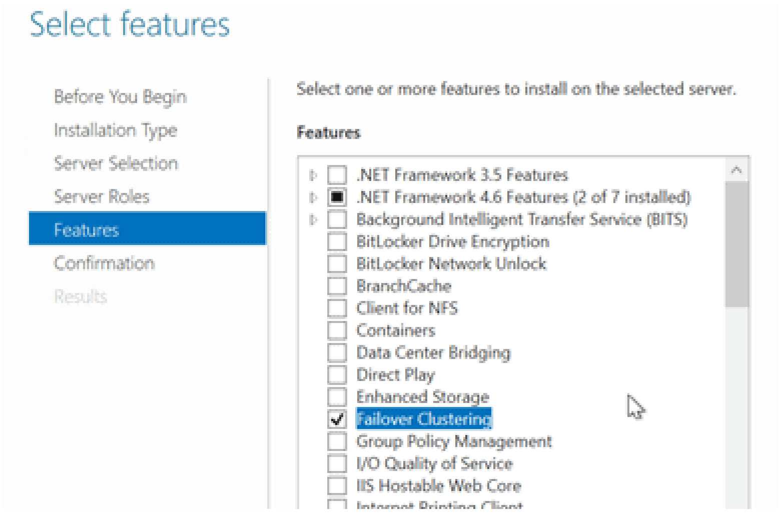
Add Windows Failover Cluster (WSFC) to each replica

Note: This task can be done by Windows Team or Infra Team

On each replica, open **Server Manager** > click **Add Roles & Features** > select **Add Failover Clustering** > click **Install.** Proceed through the wizard, and when you get to the **Select Features** page, select the **Failover Clustering** checkbox.

If you do not already have **.NET Framework 3.5.1 or greater** installed on your server, select that checkbox as well to install. (If you do need to install the .NET Framework, you will need to reboot the server after installing).

Proceed next through the wizard and click **Install** to finish the wizard. You will need to do this on every replica in your AG.



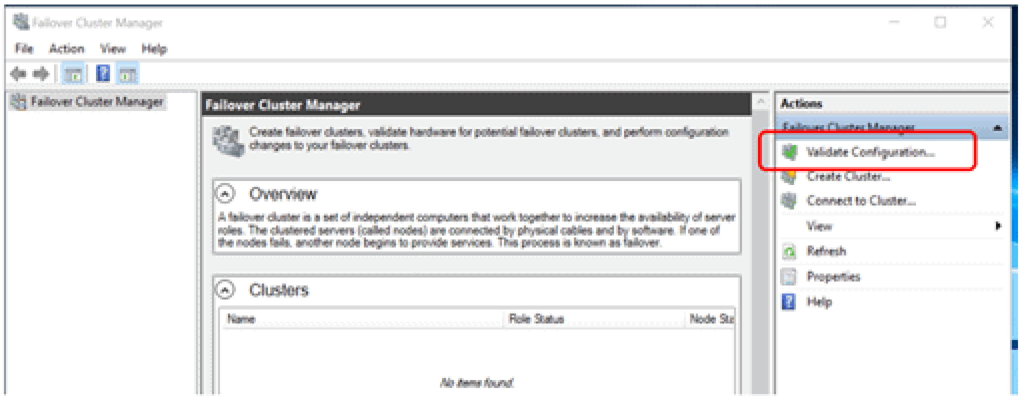
Select one or more features to install on the selected server.

Features

- [ ] .NET Framework 3.5 Features
- [■] .NET Framework 4.6 Features (2 of 7 installed)
- [ ] Background Intelligent Transfer Service (BITS)
- [ ] BitLocker Drive Encryption
- [ ] BitLocker Network Unlock
- [ ] BranchCache
- [ ] Client for NFS
- [ ] Containers
- [ ] Data Center Bridging
- [ ] Direct Play
- [ ] Enhanced Storage
- [✓] Failover Clustering
- [ ] Group Policy Management
- [ ] I/O Quality of Service
- [ ] IIS Hostable Web Core

Configure WSFC on primary replica

From Administrative Tools, open **Failover Cluster Manager** and click on **Validate Configuration**

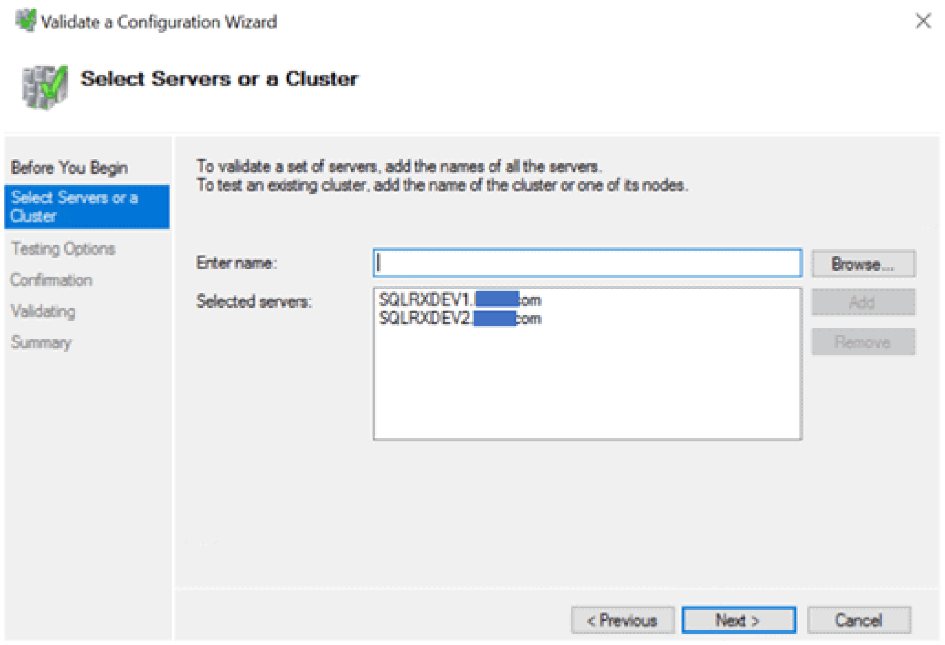


Create failover clusters, validate hardware for potential failover clusters, and perform configuration changes to your failover clusters.

A failover cluster is a set of independent computers that work together to increase the availability of server roles. The clustered servers (called nodes) are connected by physical cables and by software. If one of the nodes fails, another node begins to provide services. This process is known as failover.

Name	Role Status	Node Sta
No items found.		

Add the names of all the SQL Servers you want to configure as replicas in your AlwaysOn group.

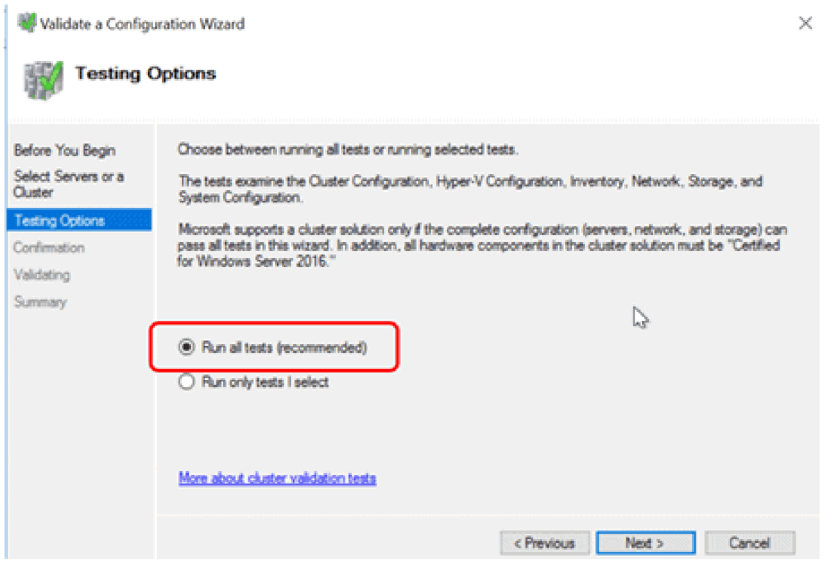


To validate a set of servers, add the names of all the servers.
To test an existing cluster, add the name of the cluster or one of its nodes.

Enter name:

Selected servers:
SQLRXDEV1.[redacted]om
SQLRXDEV2.[redacted]om

On the **Testing Options** page, click **Run all tests (recommended)**. It is normal to see some warning messages. Make sure to review the warnings and correct anything necessary.



Validate a Configuration Wizard

Testing Options

- Before You Begin
- Select Servers or a Cluster
- Testing Options
- Confirmation
- Validating
- Summary

Choose between running all tests or running selected tests.

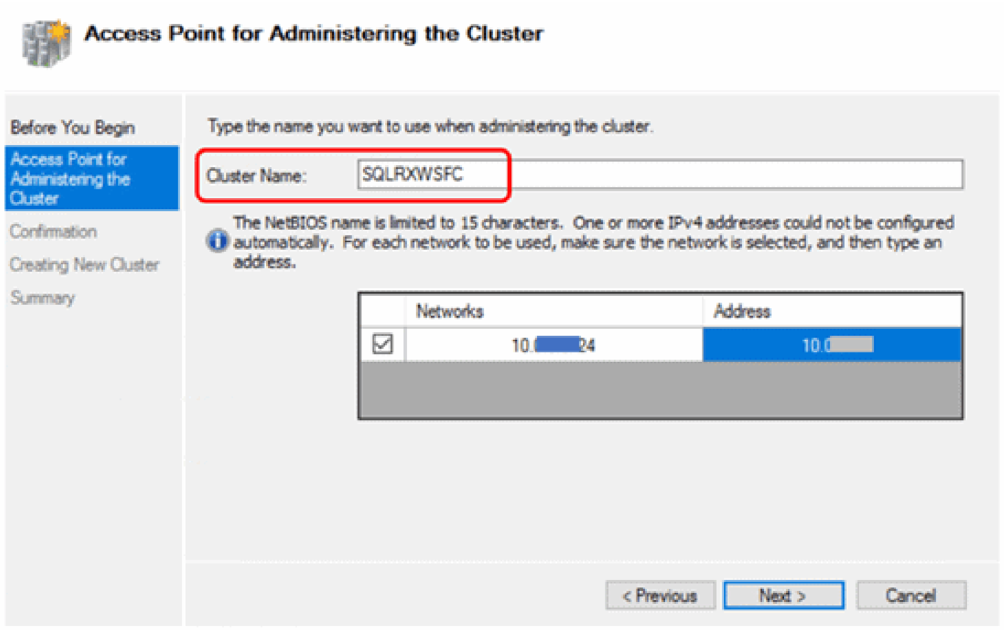
The tests examine the Cluster Configuration, Hyper-V Configuration, Inventory, Network, Storage, and System Configuration.

Microsoft supports a cluster solution only if the complete configuration (servers, network, and storage) can pass all tests in this wizard. In addition, all hardware components in the cluster solution must be "Certified for Windows Server 2016."

[✓ Run all tests (recommended)]
Run only tests I select

More about cluster validation tests

After the validation and summary is complete, the **Create Cluster Wizard** will open. In the **Access Point for Administering the Cluster** dialog box, enter the virtual cluster name (not the server or instance name), and the virtual IP address of your cluster.



Access Point for Administering the Cluster

- Before You Begin
- Access Point for Administering the Cluster
- Confirmation
- Creating New Cluster
- Summary

Type the name you want to use when administering the cluster.

Cluster Name: SQLRXWSFC

The NetBIOS name is limited to 15 characters. One or more IPv4 addresses could not be configured automatically. For each network to be used, make sure the network is selected, and then type an address.

	Networks	Address
✓	10.[redacted]24	10.[redacted]

Proceed next through the wizard, and your cluster will be created. The secondary nodes will be added to the cluster, and your cluster should now show up on all replicas (through Failover Cluster Manager). You do not have to go through these steps on the other replicas…you're all done with setting up the cluster.

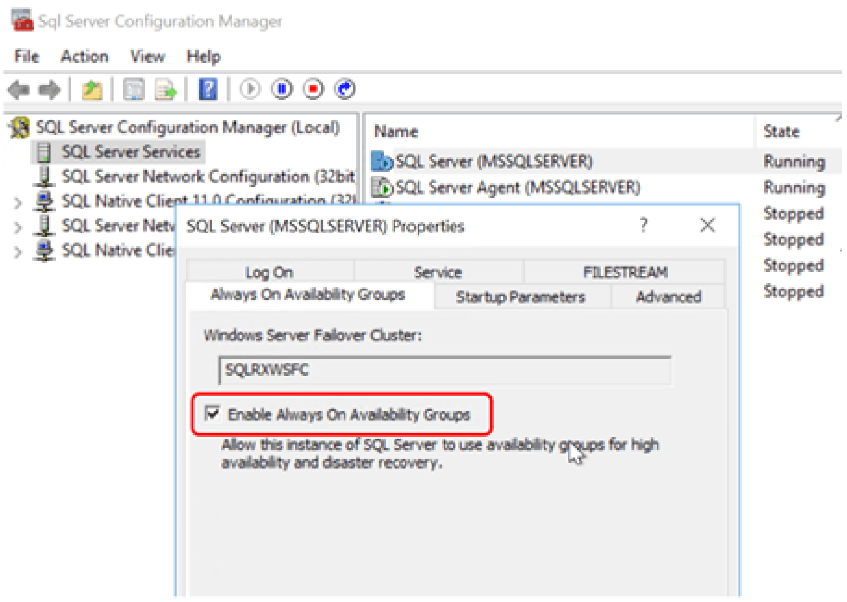
Install AlwaysOn

Note: This has to be done by DBA

To install an AG on SQL Server 2019, we need to configure the SQL Server instance and then add the AG.

Configure SQL Server

Assuming you are installing a AlwaysOn Availability Group (, make sure you have installed Enterprise Edition of SQL Server onto each replica, and install it as stand-alone instances. On each replica, open the **SQL Server Configuration Manager**. Right click on **SQL Server Services** and open the Properties dialog box. Navigate to the **AlwaysOn High Availability** tab and select the Enable AlwaysOn Availability Groups checkbox.


Sql Server Configuration Manager
File
Action
View
Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32
SQL Server Netw
SQL Native Clie
Name
State
SQL Server (MSSQLSERVER)
Running
SQL Server Agent (MSSQLSERVER)
Running
Stopped
Stopped
Stopped
Stopped
SQL Server (MSSQLSERVER) Properties
Log On
Service
FILESTREAM
Always On Availability Groups
Startup Parameters
Advanced

Windows Server Failover Cluster:
SQLRXWSFC

☑ Enable Always On Availability Groups

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

Restart the SQL Server Service after making these changes. Complete these steps on all your replicas.

Create an Availability Group

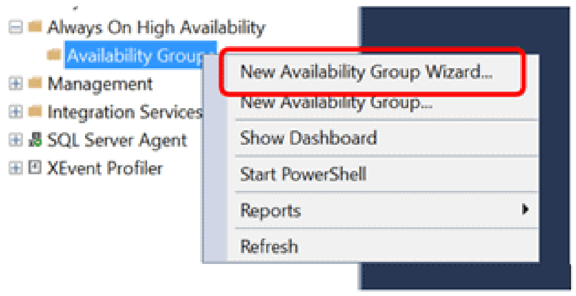
First, make sure all databases are in Full Recovery mode. Remove these databases from any transaction log backup maintenance during the installation of the Availability Group. You can add them back later. You do not want log backups happening on these databases while the AG is being created.

Next, take full and log backups of all databases you want added to the AG:

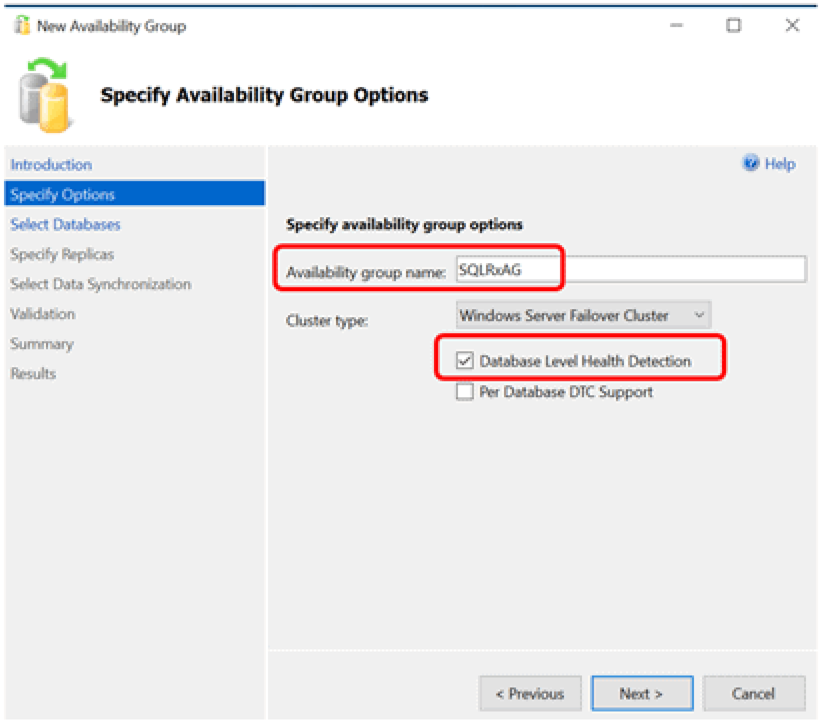
```
--Full BackupsBACKUP DATABASE [AdventureWorks2017]
```

```
TO  DISK = N'S:\MSSQL\Backups\AdventureWorks_FullBackup.bak'
WITH NOFORMAT, NOINIT, NAME = N'AdventureWorks2017-Full Database Backup',
SKIP, NOREWIND, NOUNLOAD, COMPRESSION,  STATS = 10, CHECKSUM
GO
--Tlog BackupsBACKUP LOG [AdventureWorks2017]
TO  DISK = N'S:\MSSQL\Backups\Adventureworks_TlogBackup.trn'
WITH NOFORMAT, NOINIT, NAME = N'AdventureWorks2017-Full Database Backup',
SKIP, NOREWIND, NOUNLOAD, COMPRESSION,  STATS = 10, CHECKSUM
GO
```

On your primary replica, open SQL Management Studio (SSMS) and expand the **AlwaysOn High Availability** folder. Right click on **Availability Groups** and select **New Availability Group Wizard…** to open the wizard:



First, specify your AlwaysOn group name. Name it something descriptive and unambiguous. Also, select the checkbox for **Database Level Health Detection**. Starting in 2016, failover will occur not only if the *instance* is in trouble, but also if one or more of your *databases* is in trouble. This is not a default setting however; you must specify this when creating your Availability Group.



Specify availability group options

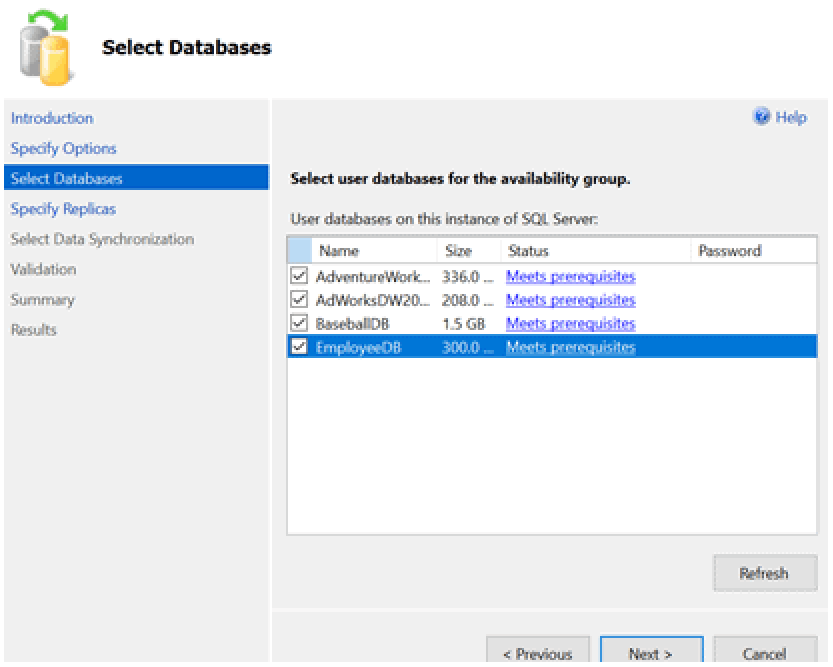
Availability group name: SQLRxAG

Cluster type: Windows Server Failover Cluster

[✓] Database Level Health Detection

[ ] Per Database DTC Support

Next, you will select the databases you want to include in your AG. All the databases in your instance will show up in this list…you don't have to include all of them in your group… select only the ones to be included in the AG.

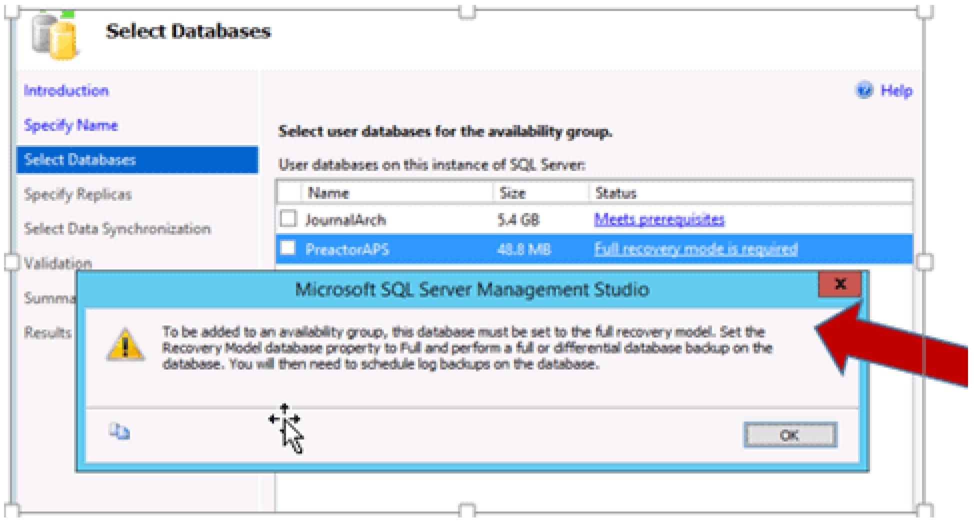


Select user databases for the availability group.

User databases on this instance of SQL Server:

	Name	Size	Status	Password
✓	AdventureWork...	336.0 ...	Meets prerequisites	
✓	AdWorksDW20...	208.0 ...	Meets prerequisites	
✓	BaseballDB	1.5 GB	Meets prerequisites	
✓	EmployeeDB	300.0 ...	Meets prerequisites	

Next to each database is a *blue link* that signifies whether your database is ready to be included into your group or not. If the link does not say 'Meets prerequisites', then you can click on the link to get a more in-depth explanation of what you need to do. Correct any discrepancies, and then select the databases to include in the AG.


Select Databases

Introduction
Specify Name
Select Databases
Specify Replicas
Select Data Synchronization
Validation
Summary
Results

Help

Select user databases for the availability group.

User databases on this instance of SQL Server:

	Name	Size	Status
☐	JournalArch	5.4 GB	Meets prerequisites
☐	PreactorAPS	48.8 MB	Full recovery mode is required

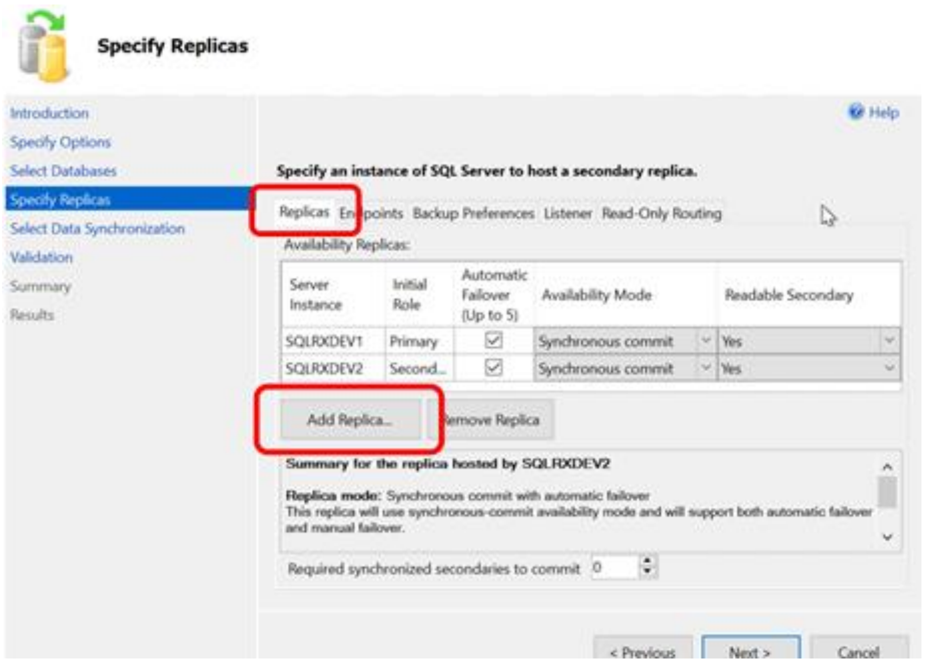
Microsoft SQL Server Management Studio

To be added to an availability group, this database must be set to the full recovery model. Set the Recovery Model database property to Full and perform a full or differential database backup on the database. You will then need to schedule log backups on the database.

OK

Next, is the **Specify Replicas** page where you will add the replicas to be included in your AG. Add and connect the replicas by clicking the **Add Replica…** button.

For each replica, you will need to specify whether you want Automatic or Manual Failover, Synchronous or Asynchronous Data Replication, and what type of connections you will allow your end users to connect with.


Specify Replicas

Introduction
Specify Options
Select Databases
Specify Replicas
Select Data Synchronization
Validation
Summary
Results

Help

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

Replicas | Endpoints | Backup Preferences | Listener | Read-Only Routing

Availability Replicas:

Server Instance	Initial Role	Automatic Failover (Up to 5)	Availability Mode	Readable Secondary
SQLRXDEV1	Primary	☑	Synchronous commit	Yes
SQLRXDEV2	Second...	☑	Synchronous commit	Yes

Add Replica... | Remove Replica

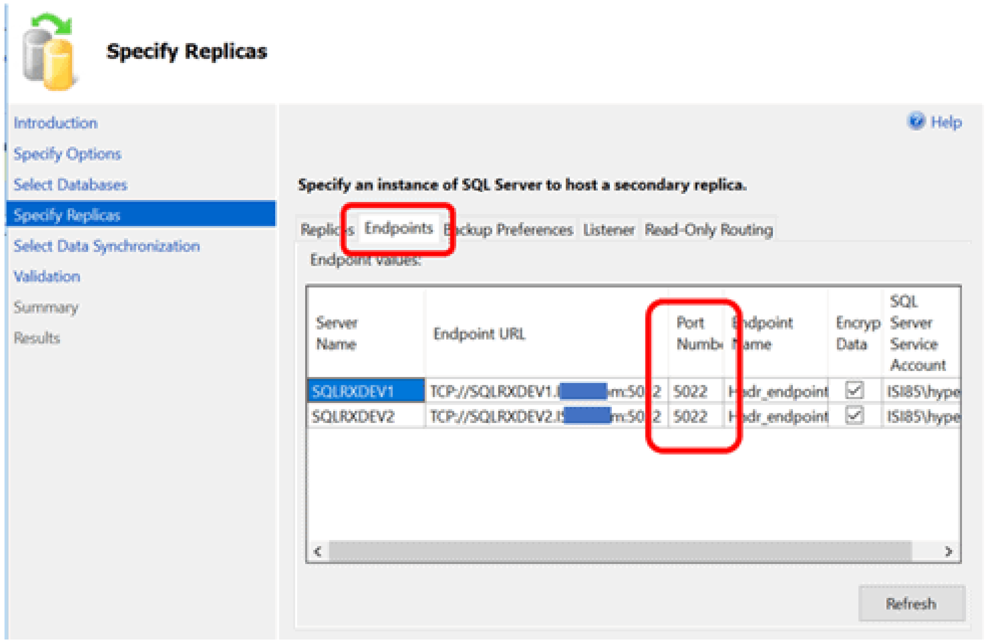
Summary for the replica hosted by SQLRXDEV2

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

Required synchronized secondaries to commit: 0

< Previous | Next > | Cancel

On this Specify Replicas page, there are several tabs at the top. The second tab is the **Endpoints** tab. On this tab verify that the port number is 5022. If you have more than one instance on your server, you might need to create another endpoint. Click here for further explanation: http://blogs.msdn.com/b/alwaysonpro/archive/2013/12/09/trouble-shoot-error.aspx

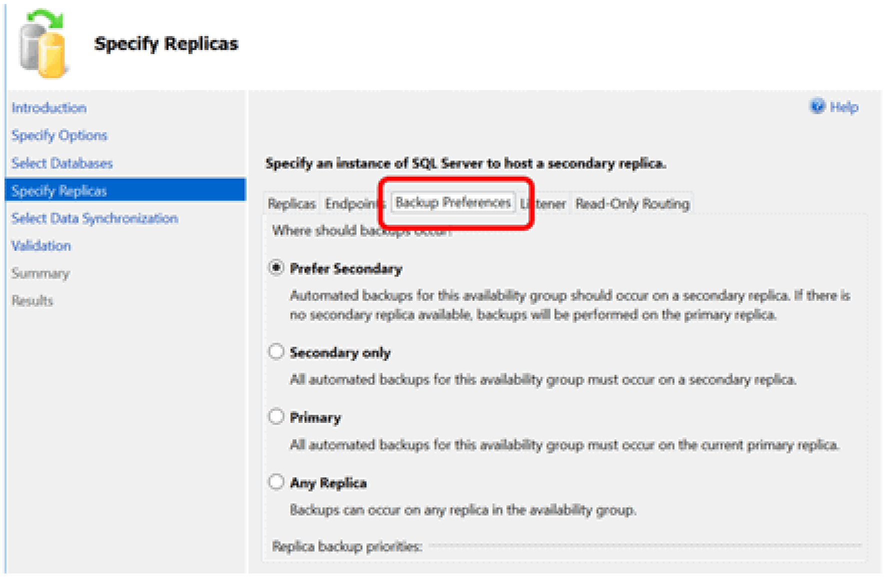

Specify Replicas

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

Endpoint values:

Server Name	Endpoint URL	Port Numb	Endpoint Name	Encryp Data	SQL Server Service Account
SQLRXDEV1	TCP://SQLRXDEV1.[redacted]m:5022	5022	Hadr_endpoint	✓	ISI85\hype
SQLRXDEV2	TCP://SQLRXDEV2.[redacted]m:5022	5022	Hadr_endpoint	✓	ISI85\hype

Next tab is the **Backup Preferences** tab. This is where you will choose where you want your backups to occur, and how you prioritize which replica will run your backups.


Specify Replicas

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

Where should backups occur?

[✓ **Prefer Secondary**]
Automated backups for this availability group should occur on a secondary replica. If there is no secondary replica available, backups will be performed on the primary replica.

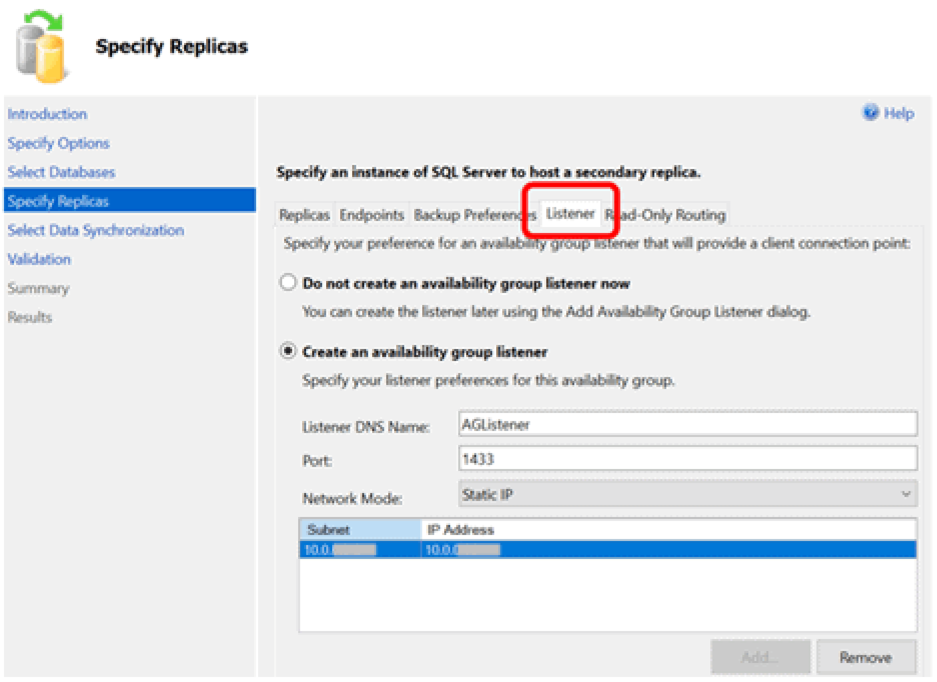
Secondary only
All automated backups for this availability group must occur on a secondary replica.

Primary
All automated backups for this availability group must occur on the current primary replica.

Any Replica
Backups can occur on any replica in the availability group.

Replica backup priorities:

The next tab in the page is the **Listener** tab. Here you will select the **Create an availability group listener** button. Enter the DNS name, which is the name that will be used in your application connection string. Enter port number 1433, and enter the IP address for your listener. This should be an unused IP address on your network.



Specify Replicas

Introduction
Specify Options
Select Databases
Specify Replicas
Select Data Synchronization
Validation
Summary
Results

Help

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

Replicas | Endpoints | Backup Preferences | Listener | Read-Only Routing

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: AGListener

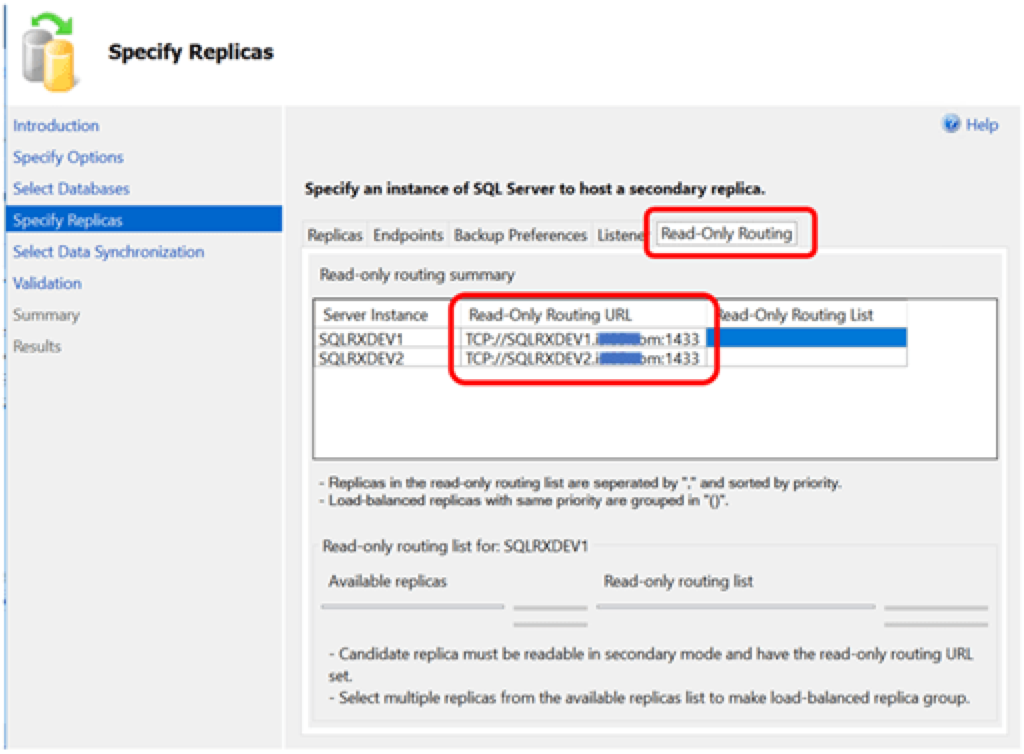
Port: 1433

Network Mode: Static IP

Subnet	IP Address
10.0.[redacted]	10.0.[redacted]

Add... Remove

The last tab in the Specify Replicas page is **Read-Only Routing**. This is *used when you want SQL Server to direct read-only connections to a secondary replica*. This feature must have a *routing URL* and a *read-only routing list*. Within the wizard you can specify the routing URL, however you cannot specify the routing list at this point. To specify the routing list, you must open the properties of the AlwaysOn group after it has been created, or add it using TSQL or PowerShell.



Specify Replicas

- Introduction
- Specify Options
- Select Databases
- Specify Replicas
- Select Data Synchronization
- Validation
- Summary
- Results

Help

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

Replicas | Endpoints | Backup Preferences | Listene | Read-Only Routing

Read-only routing summary

Server Instance	Read-Only Routing URL	Read-Only Routing List
SQLRXDEV1	TCP://SQLRXDEV1.i[redacted]om:1433	
SQLRXDEV2	TCP://SQLRXDEV2.i[redacted]om:1433	

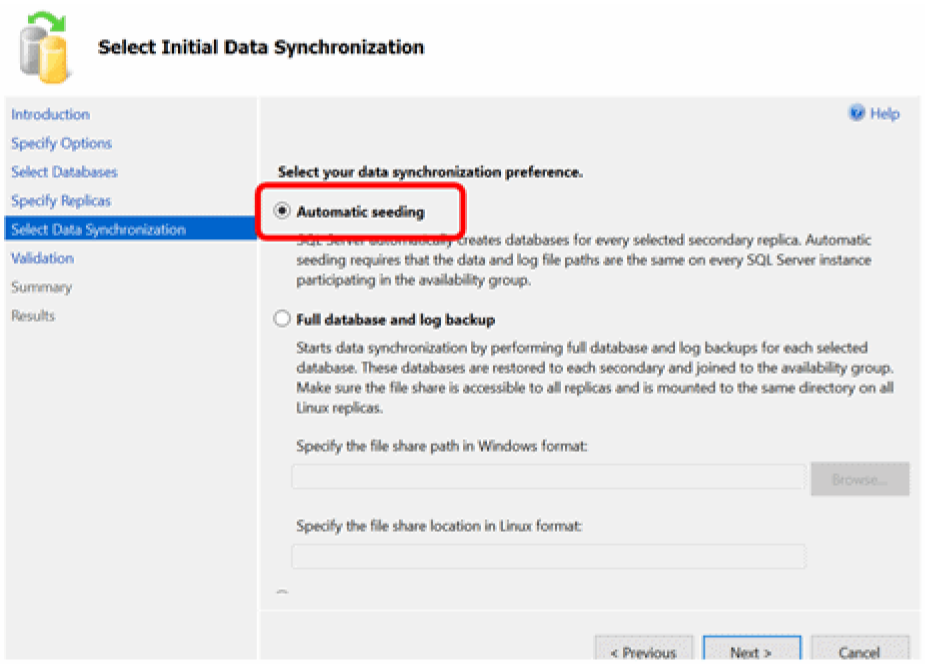
- Replicas in the read-only routing list are seperated by "," and sorted by priority.
- Load-balanced replicas with same priority are grouped in "()".

Read-only routing list for: SQLRXDEV1

Available replicas | Read-only routing list

- Candidate replica must be readable in secondary mode and have the read-only routing URL set.
- Select multiple replicas from the available replicas list to make load-balanced replica group.

The next page in the wizard is the **Select Initial Data Synchronization** page. Here is where you will join your databases to the AG. Select the **Automatic seeding** option for SQL to automatically create the databases on your secondary replicas. Make sure your data and log file paths are identical on all replicas.



Select Initial Data Synchronization

- Introduction
- Specify Options
- Select Databases
- Specify Replicas
- Select Data Synchronization
- Validation
- Summary
- Results

Help

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.

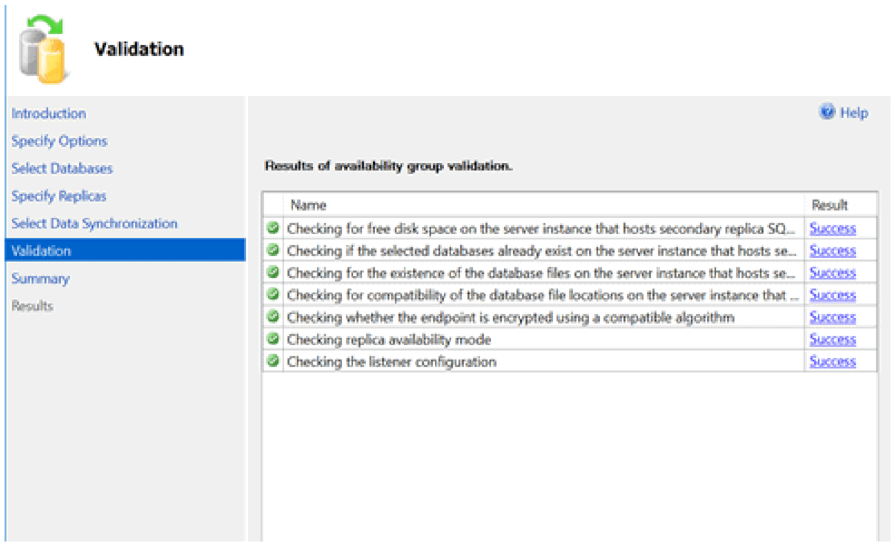
Specify the file share path in Windows format:

Browse...

Specify the file share location in Linux format:

< Previous | Next > | Cancel

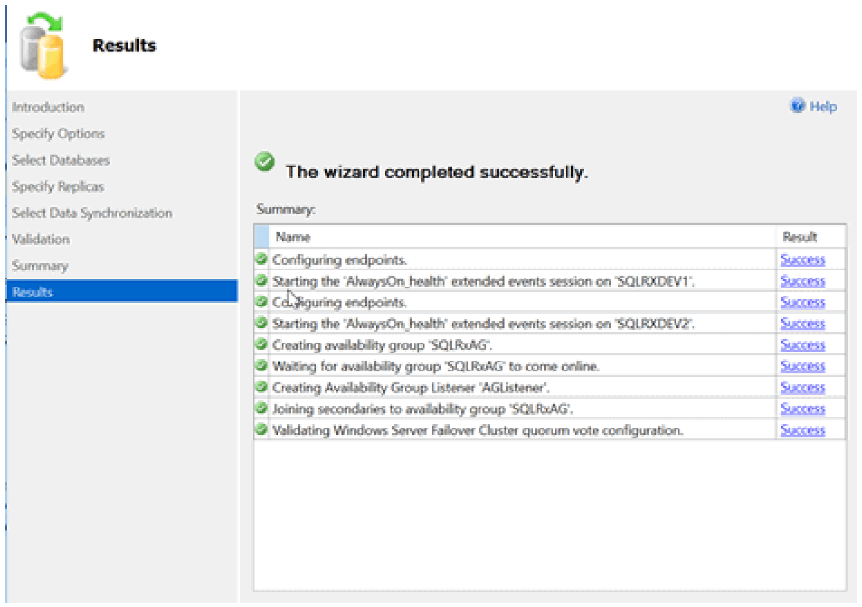
Next, ensure that your Validation checks return successful results. If you get any errors, you need to stop and correct these before proceeding.



Results of availability group validation.

Name	Result
Checking for free disk space on the server instance that hosts secondary replica SQ...	Success
Checking if the selected databases already exist on the server instance that hosts se...	Success
Checking for the existence of the database files on the server instance that hosts se...	Success
Checking for compatibility of the database file locations on the server instance that ...	Success
Checking whether the endpoint is encrypted using a compatible algorithm	Success
Checking replica availability mode	Success
Checking the listener configuration	Success

In the **Summary** page, verify that all your configuration settings are correct, and click Finish. The Results page will show the progress of the installation. Verify that all tasks have completed successfully.



The wizard completed successfully.

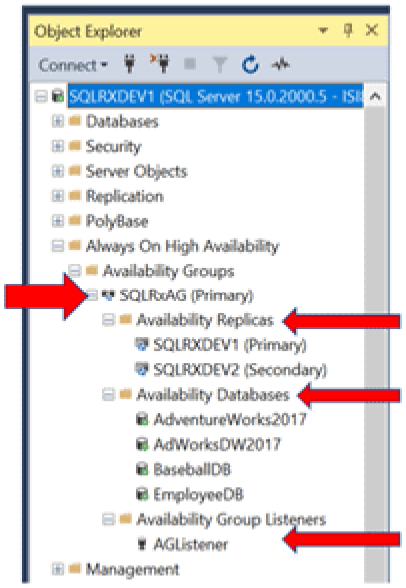
Summary:

Name	Result
Configuring endpoints.	Success
Starting the 'AlwaysOn_health' extended events session on 'SQLRXDEV1'.	Success
Configuring endpoints.	Success
Starting the 'AlwaysOn_health' extended events session on 'SQLRXDEV2'.	Success
Creating availability group 'SQLRxAG'.	Success
Waiting for availability group 'SQLRxAG' to come online.	Success
Creating Availability Group Listener 'AGListener'.	Success
Joining secondaries to availability group 'SQLRxAG'.	Success
Validating Windows Server Failover Cluster quorum vote configuration.	Success

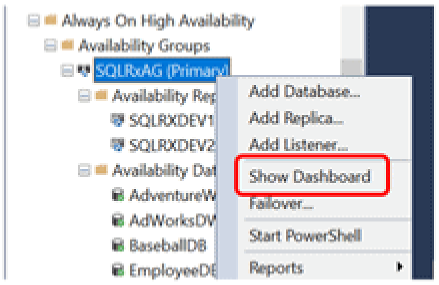
After the results are complete, and everything has finished successfully, you can now see the Availability Group created in SSMS. Based on my entries, I see these values:

- The Availability Group Name: **SQLRxAG**
- All the Replicas, and whether they are primary or secondary
- All the Databases included in the AlwaysOn group
- The Listener created for the group.

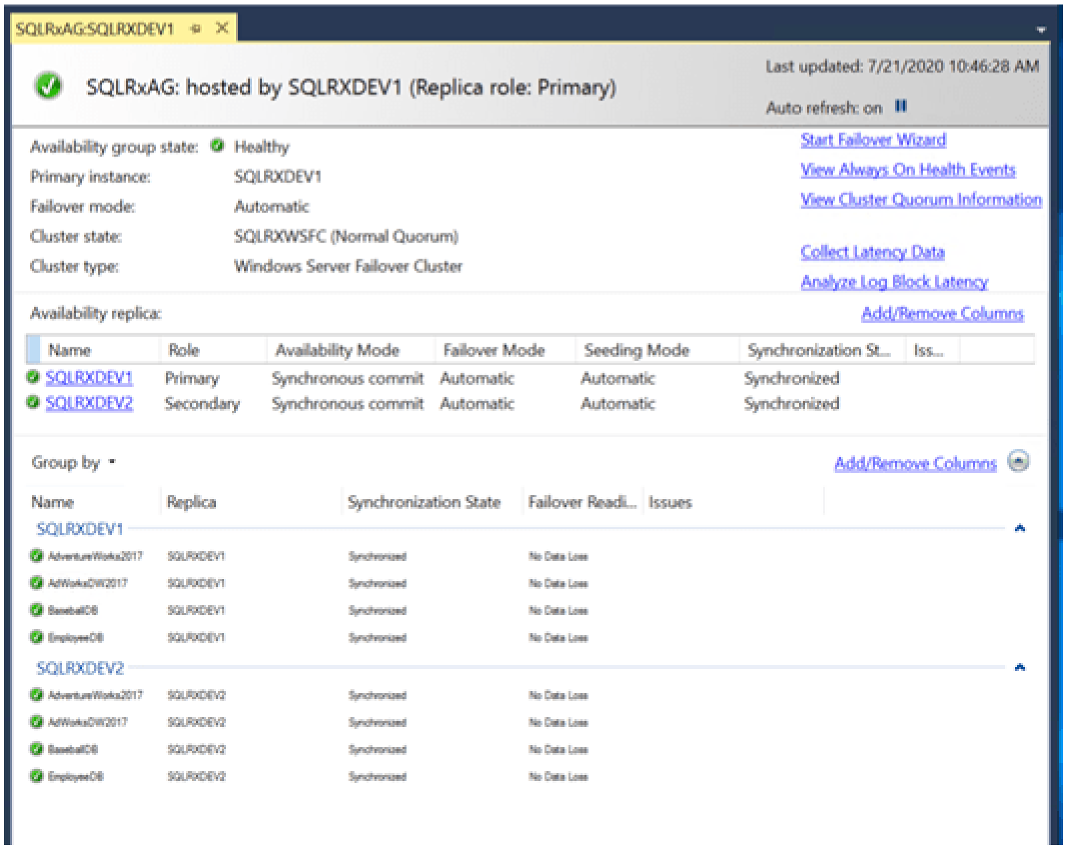
This AG will also be visible on all the secondary replicas as well.



SQL Management Studio (SSMS) provides a dashboard tool to monitor the current state and health of your Availability Groups. Simply right click on the AG and select **Show Dashboard** to get an overview of the state of your AG.



In the Dashboard, you can see which replicas are primary/secondary, the databases in the AG, the failover mode, and if they are online and connected to the AG. If there are any issues, a link will appear in the Issues column which will help you troubleshoot.


SQLRxAG:SQLRXDEV1

SQLRxAG: hosted by SQLRXDEV1 (Replica role: Primary)

Last updated: 7/21/2020 10:46:28 AM
Auto refresh: on

Availability group state:	Healthy
Primary instance:	SQLRXDEV1
Failover mode:	Automatic
Cluster state:	SQLRXWSFC (Normal Quorum)
Cluster type:	Windows Server Failover Cluster

Start Failover Wizard
View Always On Health Events
View Cluster Quorum Information
Collect Latency Data
Analyze Log Block Latency

Availability replica:

Add/Remove Columns

Name	Role	Availability Mode	Failover Mode	Seeding Mode	Synchronization St...	Iss...
SQLRXDEV1	Primary	Synchronous commit	Automatic	Automatic	Synchronized	
SQLRXDEV2	Secondary	Synchronous commit	Automatic	Automatic	Synchronized	

Group by

Add/Remove Columns

Name	Replica	Synchronization State	Failover Readi...	Issues
SQLRXDEV1				
AdventureWorks2017	SQLRXDEV1	Synchronized	No Data Loss	
AdWorksDW2017	SQLRXDEV1	Synchronized	No Data Loss	
BaseballDB	SQLRXDEV1	Synchronized	No Data Loss	
EmployeeDB	SQLRXDEV1	Synchronized	No Data Loss	
SQLRXDEV2				
AdventureWorks2017	SQLRXDEV2	Synchronized	No Data Loss	
AdWorksDW2017	SQLRXDEV2	Synchronized	No Data Loss	
BaseballDB	SQLRXDEV2	Synchronized	No Data Loss	
EmployeeDB	SQLRXDEV2	Synchronized	No Data Loss	

sql dba alwayson interview questions – 1

Q. What is AlwaysOn in SQL Server?

Ans:

AlwaysOn Availability Groups feature is a high-availability and disaster-recovery solution that provides an enterprise-level alternative to database mirroring.

Introduced in SQL Server 2012, AlwaysOn Availability Groups maximizes the availability of a set of user databases for an enterprise. An *availability group* supports a failover environment for a discrete set of user databases, known as *availability databases* that fail over together. An availability group supports a set of read-write primary databases and one to four sets of corresponding secondary databases. Optionally, secondary databases can be made available for read-only access and/or some backup operations.

Q. What are Availability Groups?

Ans:

A **container for a set of databases,** availability databases, that fails over together. Lets consider a scenario where a set of 3 databases are interlinked based on application requirement. Now we need to setup HA for these 3 databases. If we choose mirroring we need to have a separate mirroring setup for these 3 databases where as in AlwaysOn Availability Groups easier the job by grouping all these 3 databases.

Q. What are Availability Databases?
Ans:

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

Q. Which SQL/Windows Server Editions include AlwaysOn Availability Group functionality?
Ans:
SQL Server Enterprise Edition and Windows Enterprise Edition

Q. How many replicas can I have in an AlwaysOn Availability Group?
Ans:
Total 9-1 Primary and up to 8 Secondary's.

Q. How many AlwaysOn Availability Groups can be configured in Always ON?
Ans:
Up to 10 availability groups is the recommendation, but it's not enforced

Q. How many databases can be configured in an AlwaysOn Availability Group?
Ans:
Up to 100 is the recommendation, but it's not enforced

Q. What are the Restrictions on Availability Groups?
Ans:
▪ Availability replicas must be hosted by different nodes of one WSFC cluster
▪ Unique availability group name: Each availability group name must be unique on the WSFC cluster. The maximum length for an availability group name is 128 characters.
▪ Availability replicas: Each availability group supports one primary replica and up to four secondary replicas. All of the replicas can run under **asynchronous-commit mode, or up to three of them can run under synchronous-commit mode**.

n SQL Server 2022, you can configure up to **five synchronous-commit replicas** and **nine asynchronous-commit replicas** in an Always On availability group. Here's a breakdown:

- **Primary Replica**: This is the main read-write replica.
- **Secondary Replicas**: These can be configured for read-only access or as additional read-write replicas in case of failover.

For read-only access, you can configure multiple secondary replicas to offload read-only workloads from the primary replica. This helps in balancing the load and improving performance

-
- Maximum number of availability groups and availability databases per computer: The actual number of databases and availability groups you can put on a computer (VM or physical) depends on the hardware and workload, but there is no enforced limit. Microsoft has extensively tested with 10 AGs and 100 DBs per physical machine.
- Do not use the Failover Cluster Manager to manipulate availability groups.

Q What are the minimum requirements of a database to be part of the Always ON Availability Group?
Ans:

- Availability groups must be created with user databases. Systems databases can't be used.
- Databases must be read-write. Read-only databases aren't supported.
- Databases must be multiuser databases.
- Databases can't use the AUTO_CLOSE feature.
- Databases must use the **full recovery model**, and there must be a full backup available.
- A given database can only be in a single availability group, and that database can't be configured to use database mirroring.

Q. How many read-write and read only databases replica can be configure in SQL Server 2012 and 2014?
Ans:

- SQL Server 2012 supported a maximum of four secondary replicas.
- With SQL Server 2014 onwards , AlwaysOn Availability Groups now supports up to eight secondary replicas.

Q. Is it possible to setup Log Shipping on a database which is part of Availability Group?
Ans:
Yes, it can be configured.

Q. Is it possible to setup Replication on a database which is part of Availability Group?
Ans:
Yes, it is possible.

Q. Can system database participate in AG?
Ans:
No.

Q: What version of Windows do I need for AlwaysOn AGs?
Ans:
We highly recommend Windows Server 2012 or later.

Q: Can I have different indexes or tables on my replicas?
Ans:
No, the replica database contents will be exactly the same as the primary.

Q. What is Availability mode in Always ON?
Ans:
Availability mode in AlwaysOn Availability Groups refers to how transactions are committed and synchronized between the primary and secondary replicas. There are two main availability modes:

Asynchronous-commit mode: Primary replica commits the transaction on a database without waiting for the conformation from the secondary replica.
Synchronous-commit mode: Primary replica does not commit the transaction on a database until it gets the confirmation (written the transaction log records to disk on secondary) from secondary replica.
Q. Do we need SQL Server Cluster instances to configure Always ON?
Ans:
No we don't need SQL Server Cluster instances to configure Always ON.

Q. Do we need shared storage to configure Always ON?
Ans:
No, we don't need shared storage.

Q. What is the Difference between Asynchronous-commit mode and Synchronous-commit mode?
Ans:
Asynchronous-commit mode:
An availability replica that uses this availability mode is known as an asynchronous-commit replica. Under asynchronous-commit mode, the primary replica commits transactions without waiting for acknowledgement that an asynchronous-commit secondary replica has hardened the log. Asynchronous-commit mode minimizes transaction latency on the secondary databases but allows them to lag behind the primary databases, making some data loss possible.

Synchronous-commit mode:
An availability replica that uses this availability mode is known as a synchronous-commit replica. Under synchronous-commit mode, before committing transactions, a synchronous-commit primary replica waits for a synchronous-commit secondary replica to acknowledge that it has finished hardening the log. Synchronous-commit mode ensures that once a given secondary database is synchronized with the primary database, committed transactions are fully protected. This protection comes at the cost of increased transaction latency.

Q. What is called Primary replica?
Ans:

The availability replica that makes the primary databases available for read-write connections from clients is called Primary Replica. It sends transaction log records for each primary database to every secondary replica.

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

Q. What is Availability Group listener?
Ans:
Availability Group Listener is a server name to which clients can connect in order to access a database in a primary or secondary replica of an AlwaysOn availability group. Availability group listeners direct incoming connections to the primary replica or to a read-only secondary replica.

An Availability Group Listener is a virtual network name (VNN) that enables clients to connect to an AlwaysOn availability group in SQL Server without needing to know the specific IP addresses of the primary or secondary replicas. Here's how it works and why it's important:

Overview of an Availability Group Listener

1. **Virtual Network Name (VNN)**:
   - The listener is a VNN that acts as a single point of connection for applications.
   - It abstracts the underlying physical IP addresses of the replicas, making failovers transparent to the client.
2. **DNS Name**:
   - The listener has a DNS name that clients use to connect to the availability group.
   - You configure the DNS name during the creation of the listener.
3. **IP Address**:
   - The listener has one or more IP addresses associated with it, depending on the network configuration.
   - The IP address can be either static or dynamic, based on the cluster configuration.

Key Benefits

- **Automatic Redirection**: When a failover occurs, the listener automatically redirects client connections to the new primary replica, ensuring high availability without manual intervention.
- **Simplified Connection Strings**: Clients can use a single connection string with the listener's DNS name, eliminating the need to update connection strings during failovers.

- **Load Balancing**: You can configure read-only routing, allowing read-only queries to be distributed among multiple secondary replicas for better performance and load balancing.

Setting Up an Availability Group Listener

1. **Create the Listener**: You can create the listener using SQL Server Management Studio (SSMS) or Transact-SQL commands.
2. **Specify DNS Name and IP Address**: Provide the DNS name and IP address for the listener during creation.
3. **Configure Read-Only Routing**: If needed, set up read-only routing to direct read-only workloads to secondary replicas.

Example of Creating a Listener with Transact-SQL

sql

```
-- Create the Availability Group Listener
CREATE AVAILABILITY GROUP [YourAGName]
  ADD LISTENER N'YourListenerName'
  (WITH IP
    (N'IPv4', '192.168.1.10')
  );

-- Add the Listener to the Availability Group
ALTER AVAILABILITY GROUP [YourAGName]
  ADD LISTENER N'YourListenerName'
  (WITH IP
    (N
```

Q. What are Readable Secondary Replicas?
Ans:
The AlwaysOn Availability Groups active secondary capabilities include support for read-only access to one or more secondary replicas (readable secondary replicas). A readable secondary replica allows read-only access to all its secondary databases. However, readable secondary databases are not set to read-only. They are dynamic. A given secondary database changes as changes on the corresponding primary database are applied to the secondary database.

Q. What are the benefits of Readable Secondary Replicas?
Ans:
Directing read-only connections to readable secondary replicas provides the following benefits:

- Offloads your secondary read-only workloads from your primary replica, which conserves its resources for your mission critical workloads. If you have mission critical read-workload or the workload that cannot tolerate latency, you should run it on the primary.
- Improves your return on investment for the systems that host readable secondary replicas.

In addition, readable secondaries provide robust support for read-only operations, as follows:

- Temporary statistics on readable secondary database optimize read-only queries. For more information, see Statistics for Read-Only Access Databases, later in this topic.
- Read-only workloads use row versioning to remove blocking contention on the secondary databases. All queries that run against the secondary databases are automatically mapped to snapshot isolation transaction level, even when other transaction isolation levels are explicitly set. Also, all locking hints are ignored. This eliminates reader/writer contention.

Q. How many synchronous secondary replicas can I have?
Ans:
We can have up to 2 synchronous replicas, but we are not required to use any. We could run all Secondaries in asynchronous mode if desired

SQL 2022:

In SQL Server 2022, you can have up to **five synchronous-commit secondary replicas** in an Always On availability group. This means you can configure up to five secondary replicas to ensure that transactions are committed on both the primary and the secondary replicas simultaneously, providing high data protection.

Q. Can we use a secondary for reporting purpose?
Ans:
Yes. An active secondary can be used to offload read-only queries from the primary to a secondary instance in the availability group.

Q. Can we use secondary replicas to take the db backups?
Ans:

Yes, you can use secondary replicas to take database backups in SQL Server Always On availability groups. This feature is designed to offload the backup workload from the primary replica, thus improving performance and reducing the impact on the primary database. Here's how it works:

Types of Backups on Secondary Replicas

1. **Full Database Backups**: You can take full backups on secondary replicas.
2. **Log Backups**: Transaction log backups can also be performed on secondary replicas.
3. **Copy-Only Backups**: This is a special type of full backup that does not affect the differential base and is commonly used on secondary replicas.

Q. What all types of DB backups are possible on Secondary Replicas?

Ans: On secondary replicas in SQL Server Always On Availability Groups, you can perform several types of database backups. These include:

1. **Full Backups**:
   - You can take full backups of the databases on secondary replicas, which capture the entire database at a specific point in time.
2. **Differential Backups**:
   - Differential backups capture only the changes made since the last full backup. While typically performed on the primary replica, they can also be done on secondary replicas if your backup strategy allows.
3. **Transaction Log Backups**:
   - Log backups capture the transaction log entries, helping to maintain the log sequence and enabling point-in-time recovery. These can be taken on secondary replicas, reducing the load on the primary replica.
4. **Copy-Only Backups**:
   - This special type of backup does not affect the sequence of differential backups. It's commonly used for ad-hoc backups on secondary replicas without disrupting the regular backup strategy.

Q. Can we take Transaction log backups on the secondary replicas?

Ans:

Yes, we can take transaction log backups on the secondary replicas without COPY_ONLY option.

Q. What is "Failover" in Always ON?
Ans:

In the context of Always On Availability Groups, "failover" refers to the process of switching the role of a primary replica to a secondary replica. This ensures high availability and disaster recovery by allowing another replica to take over in case the primary replica becomes unavailable. There are two types of failover: automatic and manual.

Types of Failover

1. **Automatic Failover**:
   - **Trigger**: Occurs automatically in response to certain failure conditions, such as a server crash or network failure.
   - **Requirements**: Requires a synchronous-commit mode and at least one synchronous secondary replica configured for automatic failover.
   - **Impact**: Provides the highest level of availability with minimal downtime, as the system automatically redirects client connections to the new primary replica.
2. **Manual Failover**:

    - **Trigger**: Initiated manually by an administrator, typically for planned maintenance or upgrades.
    - **Requirements**: Can be performed in both synchronous-commit and asynchronous-commit modes.
    - **Impact**: Provides more control over when the failover occurs, but requires administrative intervention.

Failover Process

1. **Detection**: The system detects a failure or an administrator initiates a manual failover.
2. **Promotion**: A secondary replica is promoted to the primary role, allowing it to handle read-write operations.
3. **Reconnection**: Client connections are redirected to the new primary replica using the Availability Group Listener.
4. **Synchronization**: The former primary replica (if it becomes available again) transitions to a secondary role and resynchronizes with the new primary replica.

Example Command for Manual Failover

sql

```
ALTER AVAILABILITY GROUP [YourAGName]
  FAILOVER;
```

Benefits of Failover

- **High Availability**: Minimizes downtime by automatically or manually switching to a secondary replica.
- **Disaster Recovery**: Ensures that your database remains accessible even in the event of a failure.
- **Maintenance Flexibility**: Allows for planned maintenance and upgrades without significant impact on availability.

Q. How many types of Failover are supported by Always ON?
Ans:
Three forms of failover exist—automatic, manual, and forced (with possible data loss). The form or forms of failover supported by a given secondary replica depends on its availability mode.

Q. What are the Failover types supported by Synchronous-commit mode?
Ans:
- **Planned manual failover** (without data loss)
- **Automatic failover** (without data loss)

Q. What is planned manual failover?
Ans:
A manual failover occurs after a database administrator issues a failover command and causes a synchronized secondary replica to transition to the primary role (with guaranteed data protection) and the primary replica to transition to the secondary role.

A manual failover requires that both the primary replica and the target secondary replica are running under synchronous-commit mode, and the secondary replica must already be synchronized.

Q. What is Automatic failover?
Ans:
An automatic failover occurs in response to a failure that causes a synchronized secondary replica to transition to the primary role (with guaranteed data protection). When the former primary replica becomes available, it transitions to the secondary role. Automatic failover requires that both the primary replica and the target secondary replica are running under synchronous-commit mode with the failover mode set to "Automatic". In addition, the secondary replica must already be synchronized, have WSFC quorum, and meet the conditions specified by the flexible failover policy of the availability group.
Q. Can we configure Automatic failover of Availability Groups with SQL Server Failover cluster instances?
Ans:
SQL Server Failover Cluster Instances (FCIs) do not support automatic failover by availability groups, so any availability replica that is hosted by an FCI can only be configured for manual failover.

Q. What are the Failover types supported by under asynchronous-commit mode?
Ans:
Only form of failover is forced manual failover (with possible data loss), typically called **forced failover**. **Forced failover** is considered a form of manual failover because it can only be initiated manually. Forced failover is a disaster recovery option. It is the only form of failover that is possible when the target secondary replica is not synchronized with the primary replica.
Q. What is the use of AlwaysOn Dashboard?
Ans:
Database administrators use the AlwaysOn Dashboard to obtains an at-a-glance view the health of an AlwaysOn availability group and its availability replicas and databases in SQL Server 2012. Some of the typical uses for the AlwaysOn Dashboard are:

- Choosing a replica for a manual failover.
- Estimating data loss if you force failover.
- Evaluating data-synchronization performance.
- Evaluating the performance impact of a synchronous-commit secondary replica

Interview Questions and Answers – 2

Q. What is availability group wizard?
Ans:
Availability Group Wizard is a GUI using SQL Server Management Studio to create and configure an AlwaysOn availability group in SQL Server 2012.

Q. Suppose primary database became in suspect mode. Will AG have failover to secondary replica?
Ans:
Issues at the database level, such as a database becoming suspect due to the loss of a data file, deletion of a database, or corruption of a transaction log, do not cause an availability group to failover.

If the primary database in an Always On Availability Group (AG) enters suspect mode, the AG will not automatically failover to a secondary replica. Here's why:

Reason

- **Database-Level Issue**: A database entering suspect mode is considered a database-level issue rather than an instance-level failure. Automatic failover in an Always On AG is generally triggered by instance-level failures such as server crashes, network failures, or other critical issues affecting the availability of the SQL Server instance.

Manual Intervention Needed

- **Manual Failover**: In the case of the primary database entering suspect mode, you would need to manually initiate a failover to a secondary replica if you want to ensure availability. This manual intervention allows you to assess and address the issue with the primary database before switching roles.

Example Command for Manual Failover

sql

```
ALTER AVAILABILITY GROUP [YourAGName]
  FAILOVER;
```

After failing over, you would then troubleshoot and resolve the issue causing the primary database to enter suspect mode.

Suppose Secondary database became in suspect mode. Will AG have failover to secondary replica?

No, an Always On Availability Group (AG) will not automatically failover if a secondary database enters suspect mode. The failover mechanism in AG is designed to respond to issues affecting the primary replica or SQL Server instance, such as server crashes, network failures, or other critical issues.

Key Points

- **Primary Focus**: Automatic failover is primarily concerned with the health and availability of the primary replica.
- **Manual Intervention**: Issues with secondary databases, such as entering suspect mode, require manual intervention to resolve. These issues do not trigger a failover.

Steps to Address a Suspect Secondary Database

1. **Investigate the Issue**: Determine the cause of the suspect mode, which could be due to corruption, insufficient disk space, or other underlying problems.
2. **Restore the Database**: If necessary, restore the secondary database from a backup.
3. **Resynchronize the Replica**: Once the issue is resolved, resynchronize the secondary database with the primary replica.

By understanding and addressing issues with secondary replicas, you can maintain the overall health and availability of your Always On Availability Group.

Q. Can we have two primary availability replica?
Ans:
No, it is not possible.

No, in an Always On Availability Group, you cannot have two primary replicas. The architecture of Always On Availability Groups is designed to have one primary replica that handles read-write operations, and one or more secondary replicas that can be used for read-only operations, backups, and as failover targets.

Key Points

- **Single Primary Replica**: There is only one primary replica that handles all read-write transactions. This ensures data consistency and integrity.
- **Multiple Secondary Replicas**: You can configure multiple secondary replicas (up to eight in SQL Server 2022) for read-only access and as failover targets.
- **Automatic or Manual Failover**: In case of a failure, a secondary replica can be promoted to primary, but there will still be only one primary replica at any given time.

The single-primary model is designed to maintain the consistency and integrity of the database by ensuring that only one replica is handling write operations

Q. Does AG support automatic page repair for protection against any page corruption happens?
Ans:
Yes, It automatically takes care of the automatic page repair.

Q. How to add a secondary database from an availability group using T-SQL?
Ans:
ALTER DATABASE Db1 SET HADR AVAILABILITY GROUP = <AGName>;

To add a secondary database to an existing availability group using T-SQL, you'll need to follow a series of steps. Here's a detailed guide:

Steps to Add a Secondary Database

1. **Backup the Primary Database**: Ensure you have a full backup of the primary database and log backups to initialize the secondary database.

   sql

   ```
   BACKUP DATABASE [YourPrimaryDatabase]
   TO DISK = N'\\backupshare\YourPrimaryDatabase_FullBackup.bak';

   BACKUP LOG [YourPrimaryDatabase]
   TO DISK = N'\\backupshare\YourPrimaryDatabase_LogBackup.trn';
   ```

2. **Restore the Backup on the Secondary Replica**: Restore the full backup and subsequent log backups on the secondary replica with NORECOVERY to keep the database in a restoring state.

   sql

   ```
   RESTORE DATABASE [YourSecondaryDatabase]
   FROM DISK = N'\\backupshare\YourPrimaryDatabase_FullBackup.bak'
   WITH NORECOVERY;

   RESTORE LOG [YourSecondaryDatabase]
   FROM DISK = N'\\backupshare\YourPrimaryDatabase_LogBackup.trn'
   WITH NORECOVERY;
   ```

3. **Join the Database to the Availability Group**: Use the `ALTER DATABASE` command to join the restored database to the availability group on the secondary replica.

   sql

   ```
   ALTER DATABASE [YourSecondaryDatabase]
   SET HADR AVAILABILITY GROUP = [YourAGName];
   ```

Example Sequence of Commands

1. **Primary Replica: Backup Database**

   sql

   ```
   BACKUP DATABASE [YourPrimaryDatabase]
   TO DISK = N'\\backupshare\YourPrimaryDatabase_FullBackup.bak';

   BACKUP LOG [YourPrimaryDatabase]
   TO DISK = N'\\backupshare\YourPrimaryDatabase_LogBackup.trn';
   ```

2. **Secondary Replica: Restore Database**

   sql

   ```
   RESTORE DATABASE [YourSecondaryDatabase]
   FROM DISK = N'\\backupshare\YourPrimaryDatabase_FullBackup.bak'
   WITH NORECOVERY;

   RESTORE LOG [YourSecondaryDatabase]
   FROM DISK = N'\\backupshare\YourPrimaryDatabase_LogBackup.trn'
   WITH NORECOVERY;
   ```

3. **Secondary Replica: Join Database to Availability Group**

   sql

   ```
   ALTER DATABASE [YourSecondaryDatabase]
   SET HADR AVAILABILITY GROUP = [YourAGName];
   ```

Q. How to remove a secondary database from an availability group?
Ans:
ALTER DATABASE <DBName> SET HADR OFF;

To remove a secondary database from an Always On Availability Group using T-SQL, follow these steps:

Steps to Remove a Secondary Database

1. **Connect to the Secondary Replica**: First, connect to the SQL Server instance hosting the secondary replica.
2. **Remove the Database from the Availability Group**: Use the `ALTER DATABASE` statement to remove the secondary database from the availability group.

Example Command to Remove a Secondary Database

sql

```
ALTER DATABASE [YourSecondaryDatabase]
  SET HADR OFF;
```

Additional Steps (Optional)

- **Dropping the Database**: If you want to drop the database after removing it from the availability group, you can use the `DROP DATABASE` statement.

  sql

  ```
  DROP DATABASE [YourSecondaryDatabase];
  ```

- **Remove the Availability Group (if needed)**: If you need to remove the entire availability group, use the `ALTER AVAILABILITY GROUP` statement to remove it.

  sql

  ```
  ALTER AVAILABILITY GROUP [YourAGName] OFFLINE;
  DROP AVAILABILITY GROUP [YourAGName];
  ```

By following these steps, you can successfully remove a secondary database from an availability group.

Q. SQL Server 2012 AlwaysOn supports encryption and compression?
Ans:

Yes, SQL Server 2012 AlwaysOn Availability Groups support both encryption and compression, enhancing data security and performance.

Encryption

- **Transparent Data Encryption (TDE)**: TDE encrypts the entire database, logs, and backups, providing encryption at rest. It helps protect data by ensuring it is encrypted before it is written to disk and decrypted when read into memory.

  sql

  ```
  -- Example to enable TDE
  USE [YourDatabase];
  CREATE DATABASE ENCRYPTION KEY
  WITH ALGORITHM = AES_256
  ENCRYPTION BY SERVER CERTIFICATE [YourServerCertificate];
  ALTER DATABASE [YourDatabase] SET ENCRYPTION ON;
  ```

Compression

- **Backup Compression**: This feature reduces the size of backup files, which can improve backup performance and save storage space.

  sql

  ```
  -- Example to enable backup compression
  BACKUP DATABASE [YourDatabase]
  TO DISK = N'\\backupshare\YourDatabase_Backup.bak'
  WITH COMPRESSION;
  ```

These features allow you to enhance the security and efficiency of your AlwaysOn Availability Groups in SQL Server 2012.

SQL Server 2022 AlwaysOn supports encryption and compression?

Yes, SQL Server 2022 AlwaysOn Availability Groups support both encryption and compression, enhancing data security and performance.

Encryption

- **Always Encrypted**: This feature ensures sensitive data is encrypted within client applications and never appears as plaintext inside the SQL Server Database Engine. It uses column encryption keys and column master keys to protect data.
- **Transparent Data Encryption (TDE)**: TDE encrypts the entire database, logs, and backups, providing encryption at rest. It ensures data is encrypted before being written to disk and decrypted when read into memory.

Compression

- **Backup Compression**: This feature reduces the size of backup files, improving backup performance and saving storage space.
- **Log Stream Compression**: SQL Server 2022 compresses data streams for availability groups, reducing network traffic and optimizing performance. Compression is enabled by default for asynchronous replicas and can be configured for synchronous replicas using trace flags

Q. Does AG support Bulk-Logged recovery model?
Ans:
No, it does not.

Full Recovery Model: This is the recommended recovery model for databases in an availability group. It provides the highest level of data protection and allows for point-in-time recovery.

Q. Can a database belong to more than one availability group?
Ans:
No.

Q. What is session timeout period?
Ans:
Session-timeout period is a replica property that controls how many seconds (in seconds) that an availability replica waits for a ping response from a connected replica before considering the connection to have failed. By default, a replica waits 10 seconds for a ping response. This replica property applies only the connection between a given secondary replica and the primary replica of the availability group.

The **session timeout period** in SQL Server Always On Availability Groups specifies the amount of time (in seconds) that the primary replica waits to receive a ping or heartbeat message from a secondary replica before considering that secondary replica to be unavailable or disconnected.

Key Points

- **Default Timeout**: The default session timeout period is 10 seconds. This means that if the primary replica does not receive a heartbeat from a secondary replica within 10 seconds, it will mark that secondary replica as disconnected.
- **Configurable Value**: You can configure the session timeout period based on your specific requirements and network conditions. A shorter timeout period might be suitable for low-latency networks, while a longer timeout period might be needed for networks with higher latency or occasional interruptions.
- **Impact on Failover**: The session timeout period affects failover decisions. If a secondary replica is marked as disconnected due to a timeout, the availability group might trigger a failover if the disconnected replica was the primary or if failover conditions are met.

Example of Configuring Session Timeout Period

You can configure the session timeout period using Transact-SQL. Here's how to set the session timeout period to 20 seconds:

sql

```
ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'YourReplicaName'
  WITH (SESSION_TIMEOUT = 20);
```

Considerations

- **Network Stability**: Choose a session timeout period that balances between prompt detection of failures and tolerance for network instability.

- **Application Impact**: Ensure that the chosen timeout period aligns with your application's availability and performance requirements.

By configuring an appropriate session timeout period, you can enhance the reliability and responsiveness of your Always On Availability Groups.

Q. How to change the Session Timeout period?
Ans:

```
ALTER AVAILABILITY GROUP <AG Name>

MODIFY REPLICA ON '<Instance Name>' WITH (SESSION_TIMEOUT = 15);
```

Q. What are different synchronization preferences are available?
Ans:
As part of the availability group creation process, We have to make an exact copy of the data on the primary replica on the secondary replica. This is known as the initial data synchronization for the Availability Group.

What are different synchronization preferences are available?

In SQL Server Always On Availability Groups, you can configure synchronization preferences to specify how data is synchronized between the primary replica and secondary replicas. There are two main synchronization preferences:

1. Synchronous-Commit Mode

- **Description**: Ensures that transactions are committed on both the primary replica and at least one synchronous secondary replica before the transaction is considered committed.
- **Use Case**: This mode is ideal for high-availability scenarios where data loss cannot be tolerated, such as financial or medical systems.
- **Pros**: Provides high data protection and guarantees no data loss.
- **Cons**: Can introduce latency due to the requirement of waiting for acknowledgments from secondary replicas.

2. Asynchronous-Commit Mode

- **Description**: Transactions are committed on the primary replica without waiting for acknowledgments from secondary replicas. The data is sent asynchronously to secondary replicas.
- **Use Case**: Suitable for disaster recovery scenarios where replicas may be geographically distant, and some data loss is acceptable.
- **Pros**: Offers lower latency and better performance for the primary replica.

    - **Cons**: There is a risk of data loss in the event of a primary replica failure before the data is applied to the secondary replica.

By understanding and configuring these synchronization preferences, you can balance between performance, data protection, and disaster recovery needs based on your specific requirements.

Q. How many types of Data synchronization preference options are available in Always ON?
Ans:
There are three options- Full, Join only, or Skip initial data synchronization.

- **Full**:
    - **Description**: This option takes a full backup of the primary database and restores it on the secondary replicas. It also takes transaction log backups to ensure that the secondary databases are fully synchronized with the primary database before being joined to the availability group.
    - **Use Case**: Ideal for scenarios where you want a complete, automatic setup of secondary databases.
- **Join only**:
    - **Description**: This option is used when the secondary databases have already been restored and are ready to be joined to the availability group. You manually handle the backup and restore process, ensuring that the secondary databases are in a restoring state.
    - **Use Case**: Suitable for environments where backups are restored manually or via different methods.
- **Skip initial data synchronization**:
    - **Description**: This option skips the initial data synchronization process entirely. It requires you to manually ensure that the secondary databases are synchronized and in a restoring state before joining them to the availability group.
    - **Use Case**: Useful when you have custom processes or tools for managing backups and restores.

Q. Is it possible to run DBCC CHECKDB on secondary replicas?
Ans:
Yes.

Yes, it is possible to run **DBCC CHECKDB** on secondary replicas in Always On Availability Groups, starting from SQL Server 2016. This feature allows you to perform consistency checks on your secondary databases, reducing the performance impact on the primary replica.

Key Points

- **Consistency Check**: **DBCC CHECKDB** performs a consistency check of the entire database, ensuring that the logical and physical integrity of the database is maintained.
- **Read-Only Secondary**: You can run **DBCC CHECKDB** on read-only secondary replicas, which helps offload the workload from the primary replica.

Example Command

sql

```
DBCC CHECKDB (N'YourSecondaryDatabase') WITH NO_INFOMSGS, ALL_ERRORMSGS;
```

Running this command on a secondary replica helps maintain database integrity while minimizing the impact on the primary replica.

Q. Can I redirect the read-only connections to the secondary replica instead of Primary replica?
Ans:
Yes, we can specify the read_only intent in the connection string and add only secondaries (not the primary) to the read_only_routing list. If you want to disallow direct connections to the primary from read_only connections, then set its allow_connections to read_write.

Absolutely! In SQL Server Always On Availability Groups, you can configure **read-only routing** to redirect read-only connections to secondary replicas instead of the primary replica. This helps offload read workloads from the primary replica, improving overall performance and utilization of your secondary replicas.

Steps to Configure Read-Only Routing

1. **Configure Read-Only Routing Lists**: Define the order in which secondary replicas should be used for read-only routing.
2. **Specify Read-Only Routing URL**: Define the routing URL for each secondary replica, which tells SQL Server how to connect to each secondary replica.

Example Configuration

1. Set Up Read-Only Routing Lists

sql

```
ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'PrimaryReplica'
  WITH (SECONDARY_ROLE (ALLOW_CONNECTIONS = READ_ONLY));

ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'SecondaryReplica1'
  WITH (SECONDARY_ROLE (ALLOW_CONNECTIONS = READ_ONLY));

ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'SecondaryReplica2'
  WITH (SECONDARY_ROLE (ALLOW_CONNECTIONS = READ_ONLY));
```

2. Specify Read-Only Routing URL

sql

```
-- Primary replica routing URL
ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'PrimaryReplica'
  WITH (PRIMARY_ROLE (READ_ONLY_ROUTING_URL =
'TCP://primary_replica_domain:port'));

-- Secondary replicas routing URLs
ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'SecondaryReplica1'
  WITH (READ_ONLY_ROUTING_URL = 'TCP://secondary_replica1_domain:port');

ALTER AVAILABILITY GROUP [YourAGName]
  MODIFY REPLICA ON N'SecondaryReplica2'
  WITH (READ_ONLY_ROUTING_URL = 'TCP://secondary_replica2_domain:port');
```

Connect to the Read-Only Listener

When connecting your application to the read-only listener, make sure to use the `ApplicationIntent=ReadOnly` parameter in the connection string:

sql

```
Server=tcp:YourListenerName,1433;Database=YourDatabase;User
ID=YourUsername;Password=YourPassword;ApplicationIntent=ReadOnly;
```

By setting up read-only routing, your read-only queries can be directed to secondary replicas, optimizing the performance and availability of your SQL Server environment.

Q. If a DBA expands a data file manually on the primary, will SQL Server automatically grow the same file on secondaries?
Ans:
Yes! It will be automatically expanded on the Secondary replica.

Q. Is it possible to create additional indexes on read-only secondary replicas to improve query performance?
Ans:
No, it is not possible.

Q. Is it possible to create additional statistics on read-only secondaries to improve query performance?
Ans:
No. But we can allow SQL Server to automatically create statistics on read-only secondary replicas.

Q. Can we manually fail over to a secondary replica?
Ans:
Yes. If the secondary is in synchronous-commit mode and is set to "SYNCHRONIZED" you can manually fail over without data loss. If the secondary is not in a synchronized state then a manual failover is allowed but with possible data loss

Q. What is read intent option?
Ans:
There are two options to configure secondary replica for running read workload. The first option 'Read-intent-only' is used to provide a directive to AlwaysOn secondary replica to accept connections that have the property ApplicationIntent=ReadOnly set. The word 'intent' is important here as there is no application check made to guarantee that there are no DDL/DML operations in the application connecting with 'ReadOnly' but an assumption is made that customer will only connect read workloads.

he **read-intent** option is used in SQL Server Always On Availability Groups to direct read-only workloads to secondary replicas. This helps offload read operations from the primary replica, improving performance and optimizing resource utilization. When a connection string specifies the read-intent option, it indicates that the connection is for read-only purposes and should be routed to an available read-only secondary replica.

How to Use the Read-Intent Option

1. **Configure Read-Only Routing**: Set up read-only routing for the availability group to define how read-only connections should be directed to secondary replicas.
2. **Connection String**: Include the `ApplicationIntent=ReadOnly` parameter in the connection string to specify that the connection is for read-only purposes.

Example Connection String

sql

```
Server=tcp:YourListenerName,1433;Database=YourDatabase;User
ID=YourUsername;Password=YourPassword;ApplicationIntent=ReadOnly;
```

Benefits of Using Read-Intent Option

- **Load Balancing**: Distributes read-only queries among secondary replicas, reducing the load on the primary replica.
- **Improved Performance**: Enhances overall system performance by leveraging secondary replicas for read-only operations.
- **High Availability**: Ensures that read-only workloads remain available even if the primary replica is under heavy load.

By using the read-intent option, you can optimize the performance and availability of your SQL Server environment.

Q. Does AlwaysOn Availability Groups repair the data page corruption as Database Mirroring?
Ans:
Yes. If a corrupt page is detected, SQL Server will attempt to repair the page by getting it from another replica.

Q. What are the benefits of Always on feature?
Ans:
- Utilizing database mirroring for the data transfer over TCP/IP
- providing a combination of Synchronous and Asynchronous mirroring
- providing a logical grouping of similar databases via Availability Groups
- Creating up to four readable secondary replicas
- Allowing backups to be undertaken on a secondary replica
- Performing DBCC statements against a secondary replica
- Employing Built-in Compression & Encryption

Q. How much network bandwidth will I need?
Ans:
For a really rough estimate, sum up the amount of uncompressed transaction log backups that you generate in a 24-hour period. You'll need to push that amount of data per day across the wire. Things get trickier when you have multiple replicas – the primary pushes changes out to all replicas, so if you've got 3 replicas in your DR site, you'll need 3x the network throughput. Calculating burst requirements is much more difficult – but at least this helps you get started.

Q. What's the performance overhead of a synchronous replica?
Ans:
From the primary replica, ping the secondary, and see how long (in milliseconds) the response takes. Then run load tests on the secondary's transaction log drive and see how long writes take. That's the minimum additional time that will be added to each transaction on the primary. To reduce the impact, make sure your network is low-latency and your transaction log drive writes are fast.

Q. How far behind will my asynchronous replica be?
Ans:
The faster your network and your servers are, and the less transactional activity you have, the more up-to-date each replica will be. I've seen setups where the replicas are indistinguishable from the primary. However, I've also seen cases with underpowered replicas, slow wide area network connections, and heavy log activity (like index maintenance) where the replicas were several minutes behind.

Q. What's the difference between AGs in SQL 2012 and SQL 2014?
Ans:
SQL Server 2014's biggest improvement is that the replica's databases stay visible when the primary drops offline – as long as the underlying cluster is still up and running. If I have one primary and four secondary replicas, and I lose just my primary, the secondaries are still online servicing read-only queries. (Now, you may have difficulties connecting to them unless you're using the secondary's name, but that's another story.) Back in SQL 2012, when the primary dropped offline, all of the secondaries' copies immediately dropped offline – breaking all read-only reporting queries.

What's the difference between AGs in SQL 2019 and SQL 2022?

SQL Server 2022 introduces several enhancements and new features to Always On Availability Groups (AGs) compared to SQL Server 2019. Here are some key differences:

SQL Server 2019

1. **Basic Availability Groups**: Supports up to two replicas (one primary and one secondary) and one database per group.
2. **Advanced Features**: Includes features like automatic page repair, read-only routing, and backup on secondary replicas.
3. **Max Number of Replicas**: Supports up to eight secondary replicas.
4. **Read-Scale Availability Groups**: Allows for read-only workloads on secondary replicas.

SQL Server 2022

1. **Contained Availability Groups**: Introduces contained AGs, which include their own master and msdb databases, allowing for easier management of logins, jobs, and other instance-level objects within the AG.
2. **Azure Integration**: Enhanced integration with Azure, including support for Azure Arc-enabled SQL Managed Instances.
3. **Improved Performance**: Includes performance improvements such as log stream compression for asynchronous replicas, reducing network traffic and optimizing performance.
4. **Max Number of Replicas**: Continues to support up to eight secondary replicas.
5. **Enhanced Security**: Improved security features, including integration with Microsoft Purview for data governance and compliance.

Q: How do I monitor AlwaysOn Availability Groups?
Ans:
That's rather challenging right now. Uptime monitoring means knowing if the listener is accepting writeable connections, if it's correctly routing read-only requests to other servers, if all read-only replicas are up and running, if load is distributed between replicas the way you want, and how far each replica is running behind. Performance monitoring is even tougher – each replica has its own statistics and execution plans, so queries can run at totally different speeds on identical replicas.

Q: How does licensing work with AlwaysOn Availability Groups in SQL 2012 and 2014?
Ans:
All replicas have to have Enterprise Edition. If you run queries, backups, or DBCCs on a replica, you have to license it. For every server licensed with Software Assurance, you get one standby replica for free – but only as long as it's truly standby, and you're not doing queries, backups, or DBCCs on it.

Q: Can I use AlwaysOn Availability Groups with Standard Edition?
Ans:
Not at this time, but it's certainly something folks have been asking for since database mirroring has been deprecated.

Q: Do AlwaysOn AGs require shared storage or a SAN?
Ans:
No, you can use local storage, like cheap SSDs.

Q: Do Availability Groups require a Windows cluster?
Ans:
Yes, they're built atop Windows failover clustering. This is the same Windows feature that also enables failover clustered instances of SQL Server, but you don't have to run a failover clustered instance in order to use AlwaysOn Availability Groups.

Q: Do I need a shared quorum disk for my cluster?
Ans:
No

Q: If I fail over to an asynchronous replica, and it's behind, how do I sync up changes after the original primary comes back online?
Ans:
When I go through an AG design with a team, we talk about the work required to merge the two databases together. If it's complex (like lots of parent/child tables with identity fields, and no update datestamp field on the tables), then management agrees

to a certain amount of data loss upon failover. For example, “If we’re under fifteen minutes of data is involved, we’re just going to walk away from it.” Then we build a project plan for what it would take to actually recover >15 minutes of data, and management decides whether they want to build that tool ahead of time, or wait until disaster strikes.

SQL DBA AlwaysOn scenario based interview questions – 3

Q. We have got an alert “WSFC cluster service is offline”. What is your action plan?
Ans:
This alert is raised when the WSFC cluster is offline or in the forced quorum state. All availability groups hosted within this cluster are offline (a disaster recovery action is required).

Possible Reasons:
This issue can be caused by a cluster service issue or by the loss of the quorum in the cluster.

Possible Solutions:
Use the Cluster Administrator tool to perform the forced quorum or disaster recovery workflow. Once WFSC is started you must re-evaluate and reconfigure NodeWeight values to correctly construct a new quorum before bringing other nodes back online. Otherwise, the cluster may go back offline again.

Reestablishment may require if there are any High Availability features (Alwayson Availability Groups, Log Shipping, Database Mirroring) using on effected nodes.

Q. How to force a WSFC (Windows Server Failover Cluster) Cluster to start without a quorum?
Ans:
This can be done using

- Failover Cluster Manager
- Net.exe
- PowerShell

Here we’ll see how this can be done using FCM.

Failover Cluster Manager

- Open a Failover Cluster Manager and connect to the desired cluster node to force online.
- In the Actions pane, click Force Cluster Start, and then click Yes – Force my cluster to start.
- In the left pane, in the Failover Cluster Manager tree, click the cluster name.

- In the summary pane, confirm that the current Quorum Configuration value is: Warning: Cluster is running in ForceQuorum state.

Q. We have got an alert “Availability group is offline”. Can you explain about this warning and your action plan?

Ans:

This alert is raised when the cluster resource of the availability group is offline or the availability group does not have a primary replica.

Possible Reasons:

- The availability group is not configured with automatic failover mode. The primary replica becomes unavailable and the role of all replicas in the availability group become RESOLVING.
- The availability group is configured with automatic failover mode and does not complete successfully.
- The availability group resource in the cluster becomes offline.
- There is an automatic, manual, or forced failover in progress for the availability group.

Possible Solutions:

- If the SQL Server instance of the primary replica is down, restart the server and then verify that the availability group recovers to a healthy state.
- If the automatic failover appears to have failed, verify that the databases on the replica are synchronized with the previously known primary replica, and then failover to the primary replica. If the databases are not synchronized, select a replica with a minimum loss of data, and then recover to failover mode.
- If the resource in the cluster is offline while the instances of SQL Server appear to be healthy, use Failover Cluster Manager to check the cluster health or other cluster issues on the server. You can also use the Failover Cluster Manager to attempt to turn the availability group resource online.
- If there is a failover in progress, wait for the failover to complete.

We have got an alert “Availability group is offline”. Can you explain about this warning and your action plan?

When you receive an alert stating that an "Availability group is offline," it indicates a critical issue with the Always On Availability Group. This alert means that the availability group is not functioning as expected, and the primary replica is not available. Here are some possible causes and an action plan to address this issue:

Possible Causes

1. **Primary Replica Unavailability**: The primary replica might be down or unresponsive.
2. **Cluster Resource Offline**: The availability group resource in the Windows Server Failover Cluster (WSFC) might be offline.
3. **Connectivity Issues**: There could be connectivity issues between the replicas or with the cluster.

4. **Automatic Failover Failure**: If the availability group is configured for automatic failover, the failover might not have completed successfully.
5. **Dependent Resource Issues**: Any dependent cluster resource might have encountered a critical issue and gone offline.

Action Plan

1. **Check Primary Replica**:
   - Verify if the SQL Server instance hosting the primary replica is running.
   - Restart the SQL Server instance if it is down.
2. **Check Cluster Resource**:
   - Use the Failover Cluster Manager to check the status of the availability group resource.
   - Attempt to bring the availability group resource online if it is offline.
3. **Verify Connectivity**:
   - Ensure there are no network issues affecting connectivity between the replicas.
   - Check the cluster network settings and resolve any connectivity issues.
4. **Review Failover Configuration**:
   - If automatic failover is configured, verify that the secondary replicas are synchronized and ready for failover.
   - Manually initiate a failover if necessary.
5. **Check Dependent Resources**:
   - Verify the status of any dependent cluster resources and ensure they are online.
   - Resolve any issues with dependent resources to bring the availability group resource online.
6. **Monitor and Troubleshoot**:
   - Use the Always On Dashboard in SQL Server Management Studio (SSMS) to monitor the health of the availability group.
   - Review the SQL Server error logs and Windows Event Logs for any related errors or warnings.

Q. We have got an alert “Availability group is not ready for automatic failover”. Can you explain about this warning and your action plan?
Ans:
This alert is raised when the failover mode of the primary replica is automatic; however none of the secondary replicas in the availability group are failover ready.

Possible Reasons:
The primary replica is configured for automatic failover; however, the secondary replica is not ready for automatic failover as it might be unavailable or its data synchronization state is currently not SYNCHRONIZED.

Possible Solutions:
- Verify that at least one secondary replica is configured as automatic failover. If there is not a secondary replica configured as automatic failover, update the

configuration of a secondary replica to be the automatic failover target with synchronous commit.
- Use the policy to verify that the data is in a synchronization state and the automatic failover target is SYNCHRONIZED, and then resolve the issue at the availability replica.

Q. In your environment data inserted on Primary replica but not able to see that on secondary replica. When you check that Availability is in healthy state and in most cases data reflects in a few minutes but in this case it's didn't happen. Now you need to check for the bottleneck and fix the issue. Can you explain your views and workaround in this situation?

Ans:

Possible Reasons:

- Long-Running Active Transactions
- High Network Latency or Low Network Throughput Causes Log Build-up on the Primary Replica
- Another Reporting Workload Blocks the Redo Thread from Running
- Redo Thread Falls behind Due to Resource Contention

Possible Workaround:

- Use DBCC OPENTRAN and check if there are any oldest transactions running on primary replica and see if they can be rolled back.
- A high DMV (sys.dm_hadr_database_replica_states) value log_send_queue_size can indicate logs being held back at the primary replica. Dividing this value by log_send_rate can give you a rough estimate on how soon data can be caught up on the secondary replica.
- Check two performance objects SQL Server:Availability Replica > Flow Control Time (ms/sec) and SQL Server:Availability Replica > Flow control/sec. Multiplying these two values shows you in the last second how much time was spent waiting for flow control to clear. The longer the flow control wait time, the lower the send rate.
- When the redo thread is blocked, an extended event called sqlserver.lock_redo_blocked is generated. Additionally, you can query the DMV sys.dm_exec_request on the secondary replica to find out which session is blocking the REDO thread, and then you can take corrective action. You can let the reporting workload to finish, at which point the redo thread is unblocked. You can unblock the redo thread immediately by executing the KILL command on the blocking session ID. The following query returns the session ID of the reporting workload that is blocking the redo thread.

Transact-SQL

```
Select session_id, command, blocking_session_id, wait_time, wait_type, wait_resource

from sys.dm_exec_requests

where command = 'DB STARTUP'
```

- When Redo Thread Falls Behind Due to Resource Contention; a large reporting workload on the secondary replica has slowed down the performance of the secondary replica, and the redo thread has fallen behind. You can use the following DMV query to see how far the redo thread has fallen behind, by measuring the difference between the gap between last_redone_lsn and last_received_lsn.

Transact-SQL

```
Select recovery_lsn, truncation_lsn, last_hardened_lsn,

last_received_lsn, last_redone_lsn, last_redone_time

from sys.dm_hadr_database_replica_states.
```

If you see thread is indeed failing behind, do a proper investigation and take the help of resource governor and can control the CPU cycles

Note: Have a look at MSDN sites and try to understand these solutions because when you say possible solutions, immediately you might be asked about resolutions.

Q. You perform a forced manual failover on an availability group to an asynchronous-commit secondary replica, you find that data loss is more than your recovery point objective (RPO). Or, when you calculate the potential data loss of an asynchronous-commit secondary replica using the method in Monitor Performance for AlwaysOn Availability Groups, you find that it exceeds your RPO. What are the possible reasons that causes data loss is more than your recovery point objective?

Ans:

There are mainly two reasons:

- **High Network Latency or Low Network Throughput Causes Log Build-up on the Primary Replica.** The primary replica activates flow control on the log send when it has exceeded the maximum allowable number of unacknowledged messages sent over to the secondary replica. Until some of these messages have been acknowledged, no more log blocks can be sent to the secondary replica. Since data loss can be prevented only when they have been hardened on the secondary replica, the build-up of unsent log messages increases potential data loss.
- **Disk I/O Bottleneck Slows Down Log Hardening on the Secondary Replica.** If the log file and the data file are both mapped to the same hard disk, reporting workload with intensive reads on the data file will consume the same I/O resources needed by the log hardening operation. Slow log hardening can translate to slow acknowledgement to the primary replica, which can cause excessive activation of the flow control and long flow control wait times.

Q. After an automatic failover or a planned manual failover without data loss on an availability group, you find that the failover time exceeds your recovery time objective (RTO). Or, when you estimate the failover time of a synchronous-commit secondary replica (such as an automatic failover partner) using the method in Monitor Performance for AlwaysOn Availability Groups, you find that it exceeds your RTO. Can you explain what are the possible reasons which causes the failover time exceeds your RTO?

Ans:

- **Reporting Workload Blocks the Redo Thread from Running:** On the secondary replica, the read-only queries acquire schema stability (Sch-S) locks. These Sch-S locks can block the redo thread from acquiring schema modification (Sch-M) locks to make any DDL changes. A blocked redo thread cannot apply log records until it is unblocked. Once unblocked, it can continue to catch up to the end of log and allow the subsequent undo and failover process to proceed.
- **Redo Thread Falls Behind Due to Resource Contention:** When applying log records on the secondary replica, the redo thread reads the log records from the log disk, and then for each log record it accesses the data pages to apply the log record. The page access can be I/O bound (accessing the physical disk) if the page is not already in the buffer pool. If there is I/O bound reporting workload, the reporting workload competes for I/O resources with the redo thread and can slow down the redo thread.

Q. Let's say you have configured Automatic failover on SQL server 2012 AlwaysOn environment. An automatic failover triggered but unsuccessful in making secondary replica as PRIMARY. How do you identify that failover is not successful and what are the possible reasons that causes an unsuccessful failover?

Ans:

If an automatic failover event is not successful, the secondary replica does not successfully transition to the primary role. Therefore, the availability replica will report that this replica is in Resolving status. Additionally, the availability databases report that they are in Not Synchronizing status, and applications cannot access these databases.

Possible Reasons for Unsuccessful Failover:

- **"Maximum Failures in the Specified Period" value is exhausted**: The availability group has Windows cluster resource properties, such as the Maximum Failures in the Specified Period property. This property is used to avoid the indefinite movement of a clustered resource when multiple node failures occur.
- **Insufficient NT Authority\SYSTEM account permissions:** The SQL Server Database Engine resource DLL connects to the instance of SQL Server that is hosting the primary replica by using ODBC in order to monitor health. The logon credentials that are used for this connection are the local SQL Server NT AUTHORITY\SYSTEM login account. By default, this local login account is granted the following permissions: 1.Alter Any Availability Group, 2.Connect SQL, 3.View server state. If the NT AUTHORITY\SYSTEM login account lacks any of these permissions on the automatic failover partner (the secondary replica), then SQL Server cannot start health detection when an automatic failover occurs. Therefore, the secondary replica cannot transition to the primary role. To investigate and diagnose whether this is the cause, review the Windows cluster log.
- **The availability databases are not in a SYNCHRONIZED state:** In order to automatically fail over, all availability databases that are defined in the availability group must be in a SYNCHRONIZED state between the primary replica and the secondary replica. When an automatic failover occurs, this synchronization condition must be met in order to make sure that there is no data loss. Therefore, if one availability database in the availability group in the synchronizing or not

synchronized state, automatic failover will not successfully transition the secondary replica into the primary role.

Q. Have you ever seen the Error 41009?
Ans:
Yes! This error might occur when you try to create multiple availability groups in a SQL Server 2012 AlwaysOn failover clustering environment. This issue can be resolved by applying Cumulative Update Package 2.

Q. Let's say you added a new file to a database which is a part of AlwaysOn Availability Groups. The add file operation succeeded on primary replica but failed in secondary replica. What is the impact and how you troubleshoot?
Ans:

This might happens due to a different file path between the systems that hosts primary and secondary replica. Failed add-file operation will cause the secondary database to be suspended. This, in turn, causes the secondary replica to enter the NOT SYNCHRONIZING state.

Resolution:

- Remove the secondary database from the availability group.
- On the existing secondary database, restore a full backup of the filegroup that contains the added file to the secondary database, using WITH NORECOVERY and WITH MOVE (Specify the correct file path as per secondary).
- Back up the transaction log that contains the add-file operation on the primary database, and manually restore the log backup on the secondary database using WITH NORECOVERY and WITH MOVE. Restore the last transaction log file with NO RECOVERY.
- Rejoin the secondary database to the availability group.

Q. Can you write T-SQL statement for joining a replica to availability group? (AG name "ProAG"
Ans:
Connect to the server instance that hosts the secondary replica and issue the below statement:

ALTER AVAILABILITY GROUP ProAG JOIN;

The same operation can be done using SSMS or using Power Shell

Q. Data synchronization state for one of the availability database is not healthy. Can you tell me the possible reasons?
Ans:
If this is an asynchronous-commit availability replica, all availability databases should be in the SYNCHRONIZING state. If this is a synchronous-commit availability replica, all

availability databases should be in the SYNCHRONIZED state. This issue can be caused by the following:

- The availability replica might be disconnected.
- The data movement might be suspended.
- The database might not be accessible.
- There might be a temporary delay issue due to network latency or the load on the primary or secondary replica.

Q. Let's say we have a premium production server and it is in AlwaysOn Availability Group. You oberve that CPU utilization is hitting top at a specific time in a day. You did an RCA and found that CPU utilization reaches top and most CPU is from backup process due to backup compression is on. Now what do you suggest? Do we have any features for backup
Ans:
Yes! There is an option to perform backup from secondary replicas. We can set this from Availability Group properties we can find "Backup Preferences" and from that we can choose one of the option from:

Preferred Secondary: Backups performed on Secondary if there is no secondary configured performed from primary

Secondary Only: Backups should be done from secondary only

Primary: Must occur on Primary Replica

Any Replica: Can occur from any replica in Availability Group

Q.Is there any specific limitations if we need to perform auto backups from secondary backups?
Ans:
Yes! There are few:

- Only Copy_Only backup allowd from secondary replica
- Differential backups not allowed from secondary replica.
- Log backups can be performed from different secondary replicas but all these backups maintains a single log chain (LSN sequence). It might help in some of the situations

Q. Have you ever applied patches / CU / service packs on Alwayson Availability Groups? Did you face any issues while applying?
Ans:
Yes! I have applied CU and service packs on SQL Server 2012 SP2 Cumulative Update 4

I had a bad experience with Alwayson AG:

After CU4 applied we saw that AlwaysOn vailiabilty Gropus are in Non- Synchronizing state.

After RCA we found that there was a huge blocking between user sessions and a unknown session, CHECKPOINT with command running as “DB_STARTUP”.

Through of the MSDN SITE we found that Microsoft declared it’s a bug and the solution chosen as below:

- We had to open an outage:
- Disable Automatic Failover
- Restart the SQL Server on Primary Replica
- Re-enable automatic failover.
- This worked and fixed the issue.

Q. Can you explain any difficult issue you have faced recently on High Availability Groups?
Ans:
Sure! We are configuring AlwaysOn AG on SQL server 2014.

We have taken backup from Primary replica and restored on secondary replica

When we are trying to add secondary replica to availability group to our surprise sql server got shut down and we found the error message:

(Error: 3449, Severity: 21, State: 1.
SQL Server must shut down in order to recover a database (database ID 1). The database is either a user database that could not be shut down or a system database. Restart SQL Server. If the database fails to recover after another startup, repair or restore. SQL Trace was stopped due to server shutdown. Trace ID = ‘1’. This is an informational message only; no user action is required.)

Cause:
We did RCA and found the below.

- Service broker is enabled at Primary Replica
- We have taken a full backup from Primary Replica
- Restored on Secondary Replica where Service Broker is not enabled
- When we try to add secondary replica to AG, Service Broker is enabled, the same GUID on availability database is detected which causes an silent error 9772:
- “The Service Broker in database “<dbname>” cannot be enabled because there is already an enabled Service Broker with the same ID”.
- This results into error 3449 and shut down the sql server unexpectedly.

Solution:
This has been fixed by applying the CU1 on SQL Server 2014.

Q. Replica is in “resolving” status? What does it mean?
Ans:
A replica is into “RESOLVING” state when a auto failover is not successful.

Additionally the availability databases reports that they are in non-synchronizing state and not accessible.

Q. What are the top reasons that cause an unsuccessful failover?
Ans:

- Auto failovers in a specific period may crossed the value “Maximum Failures in the Specified Period”
- Insufficient NT Authority\SYSTEM account permissions
- The availability databases are not in a SYNCHRONIZED state