PASS SUMMIT 2015

High Availability-Disaster Recovery 101

DBA-100

Glenn Berry, Principal Consultant, SQLskills.com

PASS

Glenn Berry

- Consultant/Trainer/Speaker/Author
- Principal Consultant, SQLskills.com
  - Email: Glenn@SQLskills.com
  - Blog: http://www.SQLskills.com/blogs/Glenn
  - Twitter: @GlennAlanBerry
  - Regular presenter at worldwide conferences on hardware, scalability, and DMV queries
  - Author of SQL Server Hardware
  - Chapter author of Pro SQL Server Practices
  - Chapter author of Professional SQL Server 2012 Internals and Troubleshooting
  - Chapter author of MVP Deep Dives Volumes 1 and 2
- Instructor-led training: Immersion Events
- Online training: pluralsight hardcore developer training http://pluralsight.com/
- Consulting: health checks, hardware, performance, upgrades
- Become a SQLskills Insider: http://www.sqlskills.com/Insider

Agenda

- Causes of downtime and data loss
- Planning a high availability strategy
- SQL Server 2016 high availability technologies
- Planning a disaster recovery strategy
- SQL Server 2016 disaster recovery methods

Definition of High Availability

- Availability means that “something” is able to be used as expected
  - Example: The backend database behind a web site is able to service transactions
- High availability means that the “something” is protected by various technologies to prevent it from becoming unavailable
  - Example: The backend database is protected with database mirroring so that it continues to be available if disaster strikes
- Users/applications are always able to do what they need to be able to do
- But what is the “something” mentioned above?

What is the “Something”?

- The “something” will vary by situation, and so will the protecting technologies
- Example: a table
  - Could be protected by replication, or a solution that protects the whole database
- Example: a group of databases
  - Could be protected by an Availability Group in SQL Server 2012 or newer
- Example: a server
  - Could be protected by failover clustering
- Example: a data center
  - Could be protected using SAN replication

Causes of Downtime and Data Loss

- Planned downtime
- Unplanned downtime

Reasons for Planned Downtime

- Performing database maintenance
  - Creating or rebuilding a nonclustered index
  - Creating, dropping, or rebuilding a clustered index
  - Enterprise Edition has online index operations, that help reduce this issue
- Performing batch operations
  - Performing batch operations can cause downtime through blocking locks
- Performing an upgrade
  - Installing a SQL Server Service Pack or Cumulative Update
  - Installing Windows or Microsoft Updates
  - Updating drivers or firmware
  - Use “rolling upgrades” to minimize your planned downtime

Reasons for Unplanned Downtime

- Data center failure
  - Natural disasters, fire, power loss, failed network connectivity
- Server failure
  - Failed power supply, failed CPU, failed memory, operating system crashes
- I/O subsystem failure
  - Drive failure, a RAID controller failure, I/O subsystem software bug causing corruption
- Human error
  - Dropping a table, deleting or updating data in a table without specifying a predicate, setting a database offline, or shutting down a SQL Server instance

Planning a High Availability Strategy

- Requirements
  - Recovery Point Objective (RPO)
    - The maximum allowable data-loss when a failure occurs
  - Recovery Time Objective (RTO)
    - The maximum allowable downtime when a failure occurs
  - Context for SLA requirements
    - When specifying that a database must be available 99.99% of the time, is that 99.99% of 24x7 or is there an allowable maintenance window?

Allowable Downtime

Availability %	Downtime per Year	Downtime per Month	Downtime per Week	Downtime per Day
90%	36.5 days	72 hours	16.8 hours	2.4 hours
99%	3.65 days	7.2 hours	1.68 hours	14.4 minutes
99.9%	8.76 hours	43.8 minutes	10.1 minutes	1.44 minutes
99.99%	52.56 minutes	4.38 minutes	1.01 minutes	8.66 seconds
99.999%	5.26 minutes	25.9 seconds	6.05 seconds	864.3 milliseconds

SQL Server 2016 HA-Related Technologies

- Backup and Restore Methods
- Component Redundancy
- Windows Failover Clustering
- AlwaysOn Availability Groups
- Basic Availability Groups
- Database Mirroring
- Transactional Replication
- Peer-to-Peer Replication
- Log Shipping

Backup and Restore Methods

- Recovery models – Full, Bulk-Logged, and Simple
- Backup strategy
  - Full backups, differential backups, and log backups
  - Differential backups are very useful, but often neglected
  - Backup compression, backup checksums, mirrored backups
- Recovery strategy
  - Actually test restoring your backups and have a plan for how you will do it
  - This is often ignored!
  - Instant file initialization and backup compression can reduce restore times
  - Keeping VLF counts under control reduces recovery time portion of a restore

Using a Secondary Restore Server

- It is very common to not regularly restore database backups
  - People take regular backups, but very rarely (or never) actually restore them
  - Then, they find out in an emergency that their database backups are no good
- It is also quite common for people not to run DBCC CHECKDB
  - They are concerned about the resource usage on their production server(s)
- Consider using a "Restore Server" to restore your database backups
  - You can restore each database and then run DBCC CHECKDB on it
  - This can easily be automated. You can use an older server or new desktop machine

Component Redundancy

- It is important to have redundant components for a database server
  - This helps avoid ever having to use your HA/DR technology
- You want to eliminate single points of failure where possible
  - Multiple power supplies plugged into separate circuits
  - Multiple network ports, plugged into separate network switches
  - Appropriate RAID protection for all of your logical drives
  - Hot-swappable components can help avoid down time
  - Having some cold spares available is also a good idea

Component Redundancy vs. HA/DR

- All Microsoft HA/DR technologies have some failover duration
  - Traditional FCI must move cluster resources and start SQL Server on the new node
  - Availability groups and DBM require database property changes
  - Log shipping requires a manual failover
- It is much better to avoid some unplanned failovers with redundancy
  - Component redundancy can help avoid unplanned failovers from hardware failures
  - This improves your overall uptime statistics
- Take advantage of every possibility to make your server more robust
  - The extra hardware cost involved is usually relatively small
  - Be ready for resistance for financial reasons
  - Keep in mind that this is a database server, not a web server

Windows Failover Clustering

- SQL Server failover cluster implemented on a Windows Server failover cluster
  - Multiple nodes, one or more instances
  - Requires shared storage, which is a single point of failure
  - You can use SMB 3.0 file shares for SQL Server storage instead of a SAN
  - tempdb can be located on each node with SQL Server 2012 or newer
- Provides instance-level high availability
  - All databases, logins, Agent jobs are included
- Failover time is longer than most other technologies
  - Depends on how long crash recovery takes for each database
  - Keep your VLF counts under control

AlwaysOn Availability Groups

- Availability group contains one or more user databases that failover together
  - Requires Windows failover cluster instance, but not shared storage
  - Enterprise Edition-only feature, until SQL Server 2016
- Availability database is a database that belongs in an AG
  - Primary database is the read-write copy (limit 1)
  - Secondary database is the read-only or non-readable copy
    - Up to four on SQL Server 2012 and eight on SQL Server 2014
    - Databases must be in Full recovery model at all times
- Offers relatively fast, automatic failover
- Can offload read-only activity, but no schema changes are allowed
  - Makes it harder to use as a replacement for replication for reporting purposes

Basic Availability Groups (BAG)

- New feature in SQL Server 2016 Standard Edition
  - Basic AG enables a primary database to maintain a single replica. This replica can use either synchronous or asynchronous commit mode
  - Asynchronous commit mode is a big advantage/improvement!
- Basic Availability Group Limitations
  - Limit of two replicas (primary and secondary
  - No read access on secondary replica
  - No backups on secondary replica
  - Only one database can be in a basic availability group
  - BAG cannot be upgraded to a regular AlwaysOn AG
  - Basic availability groups are only supported on Standard Edition

Database Mirroring

- Database-level high availability, deprecated in SQL Server 2012
  - Still works in SQL Server 2016, still a good solution for many scenarios
- Principal database and mirror database, on separate instances
  - Only user databases can be mirrored
  - Databases must be in Full recovery model
- Synchronous and asynchronous modes
  - Must use synchronous mode with a witness for automatic failover
  - Asynchronous mode is only allowed in Enterprise Edition
- Database mirroring offers very fast, automatic failover
  - Only a single database, only one mirror

Transactional Replication

- Replication is a broad set of technologies that enable data to be copied and distributed between servers and then synchronized to maintain consistency
  - You can replicate the entire database or just a portion of it
- Source database is a Publisher, destination is a Subscriber
  - Log reader agent picks up all write activity from Publisher database
  - This adds some read I/O workload to the log file
  - Replication changes are temporarily stored in a Distribution database
- You can have multiple subscribers in multiple locations
  - You can add additional indexes to subscriber databases for reporting

Peer-to-Peer Replication

- Database-level protection
- A form of transactional replication that lets you have multiple, writeable copies of a database
  - These copies are often in different data centers
  - Changes are sent to each peer database, and they eventually synchronize
  - Often used for scalability purposes. HA is a secondary bonus
- May require application or database schema changes
  - Example: identity columns
- Requires Enterprise Edition

Log Shipping

- Provides database-level protection
  - Can have multiple copies in multiple locations
  - Databases must be in FULL recovery model at all times
  - Requires a manual failover (although you can write code to partially automate)
  - Some data loss is possible (since last log backup that was copied over)
- Log shipping is most commonly used for DR purposes
  - Can be used to protect against user error when you have a delayed restore
  - Can be combined with most other HA technologies
  - Does not add any extra performance overhead to primary database

High Availability Features by Edition

Feature Name	Enterprise Edition	Standard Edition	Express Edition
Partial database availability	Yes	No	No
Backup compression	Yes	Yes	No
Database snapshots	Yes	No	No
Online index operations	Yes	No	No
Log shipping	Yes	Yes	No
Transactional replication	Yes	Yes	Subscriber only
Database mirroring	Yes	Yes, synch only	Witness only
Failover clustering	Yes	Yes	No
AlwaysOn Availability Groups	Yes	No, until 2016	No

Planning a Disaster Recovery Strategy

- Designing a disaster recovery strategy is integral to designing a highly-available system
- Even with the most sophisticated redundancy, recovery from total loss of all data centers can only be done using backups
- What restores you need to be able to do depends on:
  - What needs to be brought online first
  - Data loss SLA (RPO)
  - Downtime SLA (RTO)

A Good Disaster Recovery Plan…

- Should be written by the most senior staff members who have seen the most failures
- Should be tested by the most junior staff members who may be on duty late at night
  - It won't be the most senior people on call on a holiday…
  - Consider writing it for a non-DBA to be able to follow
- Should be comprehensive
  - "Restore database from backups" isn't good enough
  - What if something goes wrong?
- Should consider human factors in a widespread disaster
- Should be tested regularly and updated after each test

More DR Planning Considerations

- Consider possible problems at each step:
  - What if the server is physically damaged?
  - What if the SAN is physically damaged?
  - What if there is no power at the data center?
  - What if there is no data center?
  - Where are the off-site backups stored?
  - What if the backups are corrupt?
  - What if key staff members are unavailable?

DR Planning: People Issues

- Who gets notified first of failures?
- Who is responsible at each phase of recovery?
- Who is the “sponsor” that can resolve disputes about progress?
- Who needs to be kept informed of progress?
- Who has to authorize a failover?
- Who is in overall command of the DR effort?
- Contact info for everyone who may become involved?
- Which other teams need to be involved for success?
- How do you confirm the application is working after DR is complete?

HA/DR Testing

- Test the solution before going into production with various failures
  - Pull out a drive, unplug a server
  - Drop a table, truncate a table
  - Unplug a network cable
  - Sometimes called “chaos monkey” testing
- Try doing a bare metal install or a full restore from backups
- What if you can’t meet your SLA requirements?
  - Push back or tweak the strategy as appropriate
  - Make sure management knows what is possible BEFORE going into production
- Perform regular real-life disaster testing IN production
  - No other way to test it for real… but easier said than done

Resources

- Whitepaper: High Availability with SQL Server 2008
  - http://bit.ly/1Xl8YEJ
- Whitepaper: Proven SQL Server Architectures for High Availability and Disaster Recovery
  - http://bit.ly/1hVibUe
- Microsoft SQL Server AlwaysOn Solutions Guide for High Availability and Disaster Recovery
  - http://bit.ly/1jBOM39

Summary

- HA/DR is much more than just using a technology or feature
  - Understand your RPO and RTO SLA requirements
  - Understand your budget and infrastructure limitations
- Make sure you have a good backup/restore strategy, regardless of your HA/DR choices
- Keep in mind that you can combine HA/DR features to have a more robust solution

Thank You