Designing High Availability Database Systems using AlwaysOn Availability Groups
Designing High Availability Database Systems using AlwaysOn Availability Groups

Anthony E. Nocentino

- Consultant and Trainer
- Founder and President of Centino Systems
  - Specialize in system architecture and performance
  - BS, MS - Computer Science (almost a PhD)
  - Friend of Redgate - 2015-16
  - Microsoft Certified Professional
- Email: aen@centinosystems.com
- Twitter: @nocentino
- Blog: www.centinosystems.com/blog
- Pluralsight Author: www.pluralsight.com

#ITDevConnections
Overview

• Discovery
• Design
  • Topology
  • Operations
• HA/DR Validation
• Application Migration
What is an Availability Group?

DC1
- DC1-SQL1
- Network
- Disk

DC2
- DC2-SQL1
- Network
- Disk

#ITDevConnections
Is it hard?

- YES!
- Impact performance
- Spinning plates
  - It’s a system, you need to be able to manage the whole thing
- Collaboration is key!
Who needs to participate?

- Networking Team
- Active Directory Team
- Server Team
- DBA Team
- Storage Team
Why Is This Important?

- Recovery Objectives
  - Recovery Point Objective - RPO
  - Recovery Time Objective - RTO
- Availability
  - How much data can we lose?
  - How fast will the system fail over?
- Performance SLA
  - Do queries have to complete in a specified amount of time?
- **Get it on paper!**

#ITDevConnections
Design Objective

• For each application establish
  • RPO
  • RTO
  • Performance SLA

<table>
<thead>
<tr>
<th>Application</th>
<th>Database</th>
<th>RPO</th>
<th>RTO</th>
<th>SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>CRM_DB1</td>
<td>1 min</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td>CRM</td>
<td>CRM_DB2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRM</td>
<td>CRM_DB3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>DM_DB1</td>
<td>1 min</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td></td>
<td>DM_DB2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Does It Really Need AGs?

• Once this phase is complete we find MANY databases don’t need AGs!
  • Good operational practices
  • Backup/Restore
  • Point in time recovery
Availability Group Design

• Group databases into Availability Groups
  • The AG is the unit of failover
• Design considerations
  • Availability requirements - HA and DR?
  • Application or database dependencies
  • Performance
  • Number of databases - soft limit of around 50
Design Objective

- Group databases into AGs

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Database</th>
<th>RPO</th>
<th>RTO</th>
<th>SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>CRM_DB1</td>
<td>1 minute</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td>AG1</td>
<td>CRM_DB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AG1</td>
<td>CRM_DB3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DM_DB1</td>
<td>1 minute</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td>AG2</td>
<td>DM_DB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Replica Placement

• Choose the location of the replicas
  • High Availability and Disaster Recovery
  • Multiple data centers?
• Number of replicas
  • Which replicas will run out of which data center?
  • Distributed active workload?
  • Read only routing? - more later
Design Objective

- AG Replica Placement

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Data Center</th>
<th>Replica</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>DC1</td>
<td>DC1-SQL1</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC1</td>
<td>DC1-SQL2</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC2</td>
<td>DC2-SQL1</td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DC1</td>
<td>DC1-SQL1</td>
</tr>
<tr>
<td></td>
<td>AG2</td>
<td>DC1</td>
<td>DC1-SQL2</td>
</tr>
</tbody>
</table>
Designing High Availability Database Systems using AlwaysOn Availability Groups

DC1

CRM Availability Group – AG1

Document Management – AG2

DC1-SQL1
DC1-SQL2

DC2

DC2-SQL1
Establish Quorum Model

- AGs still use Windows Failover Clustering
  - 2012 R2 > 2008 R2
    - Dynamic quorum - periodically recalculates based on online voters
- Given the agreed upon replica placement, define a quorum model
  - Network topology
  - Multiple sites?
  - Where is the workload going to run?
- Node majority - useful if there are odd number of nodes
- File share witness - useful if there are an even number of nodes (Microsoft Rec’d)
  - Required for multi-site, choose a third location for file share
Designing High Availability Database Systems using AlwaysOn Availability Groups

Node Majority + Witness
Design Objective

- Establish Quorum Model and Voters

<table>
<thead>
<tr>
<th>Servers</th>
<th>Quorum</th>
<th>Voter</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC1-SQL1</td>
<td>Node Majority + File Share</td>
<td>Yes</td>
</tr>
<tr>
<td>DC1-SQL2</td>
<td>Node Majority + File Share</td>
<td>Yes</td>
</tr>
<tr>
<td>DC2-SQL1</td>
<td>Node Majority + File Share</td>
<td>No</td>
</tr>
<tr>
<td>DC1-FILE1</td>
<td>Node Majority + File Share</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Designing High Availability Database Systems using AlwaysOn Availability Groups

Node Majority + Witness
Designing High Availability Database Systems using AlwaysOn Availability Groups

Node Majority + Witness
Designing High Availability Database Systems using AlwaysOn Availability Groups

Node Majority + Witness
Failover Model

- The Availability Group is the unit of failover
  - For each Availability Group and for each replica
  - Establish a failover policy
    - Automatic
    - Manual
Design Objective

- Establish Failover Model

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Replica</th>
<th>Failover</th>
<th>RPO</th>
<th>RTO</th>
<th>SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>DC1-SQL1</td>
<td>Automatic</td>
<td>1 min</td>
<td>5 min</td>
<td>10 ms</td>
</tr>
<tr>
<td>AG1</td>
<td>DC1-SQL2</td>
<td>Automatic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AG1</td>
<td>DC2-SQL1</td>
<td>Manual</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DC1-SQL1</td>
<td>Automatic</td>
<td>1 min</td>
<td>5 min</td>
<td>10 ms</td>
</tr>
<tr>
<td>AG2</td>
<td>DC1-SQL2</td>
<td>Automatic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data Movement In Availability Groups

• Transaction log blocks are replicated to secondaries
• Replication mode
  • Synchronous
    • Required for automatic failover
    • Acknowledgements are sent from secondary to primary
    • Monitor replication latency carefully
  • Asynchronous
• Database mirroring endpoint
Data Movement In Availability Groups

- You can experience data loss in both **synchronous** and **asynchronous** modes
  - Due to replication latency!
Data Movement - Send Queue

- Queues log blocks to be sent to the secondaries
- Each replica maintains its own view of the send queue
- Queued data is at risk to data loss in the event of a primary failure
- The send queue can grow due to an unreachable secondary, network outage, network latency, and large amount of data change
Data Movement - Redo Queue

- Queues log blocks received on the secondary
- Each replica has its own redo queue
- On failover, the redo queue must be completely processed
- The redo queue can grow due to a slow disk subsystem or resource contention or sustained outage and subsequent reconnection of a secondary
Design Objective

- Establish Availability Mode

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Replica</th>
<th>Failover</th>
<th>Availability Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>DC1-SQL1</td>
<td>Automatic</td>
<td>Sync</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC1-SQL2</td>
<td>Automatic</td>
<td>Sync</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC2-SQL1</td>
<td>Manual</td>
<td>Async</td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DC1-SQL1</td>
<td>Automatic</td>
<td>Sync</td>
</tr>
<tr>
<td></td>
<td>AG2</td>
<td>DC1-SQL2</td>
<td>Automatic</td>
<td>Sync</td>
</tr>
</tbody>
</table>
Designing High Availability Database Systems using AlwaysOn Availability Groups

Availability Mode
Transaction Log Throughput

• For each database
  • What is the amount of transaction log generated?
  • Include days when maintenance or large batch transactions run
  • Add that up for each DB in each AG, this will be your network bandwidth requirements

• Good - We can use compressed log backup size as an approximation of log throughput.
  • 2016 uncompressed to sync, compressed to async

• Better - If there’s a monitoring package, review data or baseline a representative workload
  • Analyze Log Bytes Flushed/sec and Log Flushes/sec

• Best - replay of a representative workload into an AG
  • Primary - Log Bytes Flushed/sec, Bytes Sent to Replica/sec (c), Network Interface
  • Secondaries - Bytes from Replica/sec (c), Log Bytes Received/Sec, Redone Bytes/sec, Network Interface
Networking

- Bandwidth analysis for both local and remote replication
  - Log blocks are what’s replicated
    - 2012/2014 - compressed
    - 2016 - uncompressed to sync, compressed to async
  - Replication to each replica
  - LAN - shared/dedicated
  - WAN
- Redundant network interfaces and uplinks for each server
- What type of network interconnects?
Design Objective

- Establish Per Replica Log Throughput

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Database</th>
<th>Replica</th>
<th>Log Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>CRM_DB1</td>
<td>DC1-SQL1</td>
<td>8Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB1</td>
<td>DC1-SQL2</td>
<td>8Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB1</td>
<td>DC2-SQL1</td>
<td>8Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB2</td>
<td>DC1-SQL1</td>
<td>2Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB2</td>
<td>DC1-SQL2</td>
<td>2Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB2</td>
<td>DC2-SQL1</td>
<td>2Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB3</td>
<td>DC1-SQL1</td>
<td>20Kb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB3</td>
<td>DC1-SQL2</td>
<td>20Kb/sec</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>CRM_DB3</td>
<td>DC2-SQL1</td>
<td>20Kb/sec</td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DM_DB1</td>
<td>DC1-SQL1</td>
<td>1Mb/sec</td>
</tr>
<tr>
<td></td>
<td>AG2</td>
<td>DM_DB2</td>
<td>DC1-SQL2</td>
<td>1Mb/sec</td>
</tr>
</tbody>
</table>
IP Addressing Requirements

• Provision static IP address for cluster management
• Provision static IP, Port on each subnet you have replicas for each AG
• DNS name for each AG listener
  • Names alphanumeric including dashes and underscores
  • DNS Aliases
Application Connectivity

- Applications connect to the Availability Group Listener
- How do applications connect to the databases
  - .NET data provider, SQL Native Client, ODBC, JDBC
Application Connectivity

• Multi-subnet failover
  • MultiSubnetFailover=True
  • RegisterAllProviderIP
• Non .NET or < .NET4
  • Adjust TTL on DNS record (A or CNAME)
Readable Secondaries

• Will there be readable secondaries?
  • Where?
• Load balancing
  • 2012/2014 - sequential list (hardware load balancer)
  • 2016 - Round robin
  • Restricting workload to the “active” site
• ApplicationIntent=ReadOnly

#ITDevConnections
Readable Secondaries (con't)

- Requires row versioning on secondary, uses RCSI
  - Monitor usage and disk pressure of TempDB on secondary
  - Additional 14 bytes on the row for versioning info
- Create supporting indexes on the primary
- Blocking of REDO on secondary can occur during schema changes
  - sch-m, sch-s

#ITDevConnections
Readable Secondaries (con't)

- A configured listener
- At least one replica is configure for read-only access
- Each secondary is configured with a URL
- Each replica has a configured routing list
- The replica being routed to must be synchronized or synchronizing

```sql
ALTER AVAILABILITY GROUP [AG1]
MODIFY REPLICA ON N'SQL14-A'
WITH (PRIMARY_ROLE(READ_ONLY_ROUTING_LIST=(N'SQL14-B',N'SQL14-A')))

ALTER AVAILABILITY GROUP [AG1]
MODIFY REPLICA ON N'SQL14-B'
WITH (PRIMARY_ROLE(READ_ONLY_ROUTING_LIST=(N'SQL14-A',N'SQL14-B')))```
Design Objective

- For each replica establish Readable Secondaries

<table>
<thead>
<tr>
<th>Application</th>
<th>Availability Group</th>
<th>Replica 1</th>
<th>Readable</th>
<th>RPO</th>
<th>RTO</th>
<th>SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRM</td>
<td>AG1</td>
<td>DC1-SQL1</td>
<td>Yes</td>
<td>1 minute</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC1-SQL2</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AG1</td>
<td>DC2-SQL1</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOC MGMT</td>
<td>AG2</td>
<td>DC1-SQL1</td>
<td>Yes</td>
<td>1 minute</td>
<td>5 minutes</td>
<td>10 ms</td>
</tr>
<tr>
<td></td>
<td>AG2</td>
<td>DC1-SQL2</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Designing High Availability Database Systems using AlwaysOn Availability Groups

DC1

- ag1.domain.local
- CRM Availability Group – AG1
- ag12.domain.local
- Document Management Availability Group – AG2

DC1 SQL1
Synchronous Automatic
Readable Secondary

DC1 SQL2
Synchronous Automatic
Readable Secondary

Client Application

DC1-FILE1
File Share Witness

DC2

- DC2-SQL1
Asynchronous Manual
Not Readable

IT/Dev Connections
Backups!

- Required FULL recovery model
- Availability Groups are only part of the HA/DR plan
- Review the current database backup scheme
  - Current backup software (this can get hairy!)
  - Current backup routine
- Review the current enterprise backup scheme
  - Replication and archiving of backups
- Offloaded backups
  - If availability and recovery are important to you, backup on the primary!

#ITDevConnections
VLDBs

- Large Tables
- Poor Indexing Strategies
- Special backup considerations
  - Differentials
- Special networking considerations
  - Dedicated networking for replication
  - QOS between sites
Application Compatibility

• Does your vendor support AGs?
• Are you using?
  • Cross Database Transactions - No!
  • Distributed Transaction Coordinator
  • Transparent Data Encryption - Painful
Database Objects

• Synchronization is up to you!
• SQL Agent Jobs
  • You’ll need to build AG aware jobs
• Database logins (ensure the SIDs are the same)
• Linked servers
Designing High Availability Database Systems using AlwaysOn Availability Groups

Operations

• Database maintenance
  • Index maintenance
  • Smart indexing
  • May need to increase the fragmentation thresholds
  • Minimize log generation
  • Can we reindex more frequently?
  • Fill factor
• Statistics maintenance
• CHECKDB
  • All replicas if possible - easiest
  • Where you take backups or any replica that could become a primary
Operations (con’t)

• Non-Production testing
  • Need to have at least one production-like environment for testing

• Patching
  • Manual failover targets, active failover targets, then primary

• Scheduled downtime
  • Anything that makes a secondary unreachable
    • Network maintenance
    • Server maintenance
  • Agree on a schedule with operations team

• Reseeding a replica
Monitoring

- Monitoring and Trending
  - Establish a baseline for analysis
    - Are we meeting recovery and performance objectives?
    - Measure impact on resources
- What do we want to do for monitoring?
  - Roll your own
  - Third party package
  - SSMS AlwaysOn Dashboard
Monitoring (con’t)

- Network throughput
- Page splits
- Log Bytes Flushed/sec and Log Flushes/sec
- Send and redo queue size
- Send and redo throughput
- Send and redo latency
- Transaction Delay
- Failover
- Listener online
Hardware

• Physical placement of servers
  • Rack location
  • Power supply

• Servers
  • Can’t use last year's hardware for secondaries or DR
  • Physical (CPU/Memory)
  • Virtualization (vCPU/Memory)

• Storage
  • Design for performance on all replica
  • No more using last years SAN or servers at DR
Disk Topology

- Design like any other tier 1 system
  - Establish a performance SLA and design to meet that
- RAID types
- Operating System
- Databases
- Logs
- System databases
- TempDB
  - TempDB configuration
Operating Systems

- Windows 2012 R2 (yes, please)
- Windows 2008 R2 (no, thanks)
  - Special circumstances for quorum
  - Becoming less of an issue
- Review base configuration of operating system
  - Power Management
  - Lock Pages in Memory
  - Instant File Initialization
  - Partition Alignment
  - 64k NTFS Allocation Units
Active Directory

- In 2012 and 2014 Active Directory is required
  - 2016 has domain-less and inter-domain clusters

- The user creating the cluster will need
  - Create computer account on OU servers are in
- Cluster Named Object - (CNO) will need
  - Create computer account on OU servers are in

- SQL Service Accounts
  - Easy - shared domain user per Windows cluster
  - Managed service accounts - not supported but work

#ITDevConnections
HA/DR Testing

• Planned failover
  • Within a data center
  • Between data centers

• Unplanned failover
  • Within a data center
  • Between data centers

• Did your applications reconnect? In time?
HA/DR Testing

- Planned failover
  - Within a data center
  - Between data centers
    - Change from async to sync then failover
    - Move quorum?
    - Backups - take a full backup
    - Did your applications connect?...in time?
HA/DR Testing (con’t)

• Unplanned failover
  • Within a data center
  • Between data centers
    • How much data did we loose?
    • Who decides when to failover?
    • Did your applications connect?…in time?
    • Move quorum?
    • Backups - take a full backup
    • Reseeding replicas
Designing High Availability Database Systems using AlwaysOn Availability Groups
Application Migration

- On-boarding of applications into the new environment
- Construct the new environment
  - Migrate databases onto the new environment
  - Add databases to Availability Groups
  - Use DNS aliases to manage the transition
Licensing

• How many replicas?

• Which secondaries are used for “SQL Workloads”?
  • Basic rule is, if you’re connecting to the replica…it needs a license

• Second replica
  • If not used for anything other than failover and on premises (not cloud)
    • For free 2012
    • Free only with SA on 2014+

• Additional replicas
  • Require license


#ITDevConnections
Designing High Availability Database Systems using AlwaysOn Availability Groups

SQL Server 2016 Enhancements

• Basic Availability Groups
• Distributed Availability Groups
  • Easier quorum designs
  • Less pressure on inter-AG networks (WAN)
• Parallel redo
• Direct Seeding

#ITDevConnections
Review

- Availability Group topology
- Application connectivity
- Operations
  - Backup
  - Monitoring
  - System and network maintenance
- It’s hard!
  - Design
  - Test
- Review the hidden slides in this deck for deeper details and more info!
Thank you

- www.centinosytems.com/blog/talks
- Links to resources
- This presentation
- The design spreadsheet
Rate This Session Now!

Tell Us What You Thought of This Session

Rate with Mobile App:
• Select the session from the Agenda or Speakers menus
• Select the Actions tab
• Click Rate Session

Rate with Website:
Register at www.devconnections.com/logintoratesession
Go to www.devconnections.com/ratesession
Select this session from the list and rate it

Be Entered to WIN Prizes!

#ITDevConnections
Thank you!

Questions?