

SQLskills Immersion Event IE3: High Availability & Disaster Recovery

Module 8: Transactional Replication

Joe Sack
Joe@SQLskills.com



Overview

- Performance considerations
- Combining replication with other technologies
- Monitoring
- Troubleshooting



2

© SQLskills, All rights reserved.
<http://www.SQLskills.com>



Replication

Terminology: Publishing Industry Metaphor

<i>Publisher</i>	{	Database instance that makes data available to other locations through replication
<i>Distributor</i>	{	Database instance that acts as a store for replication specific data associated with one or more Publishers. Distribution database stores replication status data, metadata about the publication, and, in some cases, acts as a queue for data moving from the Publisher to the Subscribers
<i>Subscriber</i>	{	Database instance that receives replicated data
<i>Publication</i>	{	Collection of one or more articles from one database
<i>Article</i>	{	Database object that is included in a publication
<i>Subscription</i>	{	Copy of a publication that is delivered to a Subscriber

Transactional Replication

- Under best deployment conditions can accommodate high throughput of transactions from server-to-server
 - Streaming changes within seconds
 - Several factors impacting latency (we'll cover)
- Data warehousing and reporting
- Integrating data from multiple sites or heterogeneous data
- Offloading batch processing

Distribution Database

- Think of it no differently from other application databases or tempdb when it comes to growth, monitoring, performance overhead
 - Transactional replication: all transactions written to and read from the distribution database
 - Merge replication: usage is significantly less, storing history and miscellaneous information about merge replication
- Database volume, availability and collocation matters and contributes to the 'local vs. remote' decision
- Separate distribution databases, on the same SQL Server instance, may be advisable for highly active publications
- Sizing can be significantly impacted by retention settings and also the immediate_sync option when 'true'

Hidden Overhead `sp_addpublication's immediate_sync`

- When `immediate_sync = 1`:
 - New Subscribers can use the latest snapshot files within the retention period
 - Commands and transactions will also be available within the retention period, increasing the distribution database size requirements significantly for databases with high throughput
 - Snapshot files for the publication are created for all articles each time the Snapshot Agent runs, even if only one article was added
- When `immediate_sync = 0`:
 - Snapshot files are only created if there are new subscriptions
 - If you add a new article and you execute the Snapshot Agent, only a snapshot of the new article is produced
- The Wizard makes it easy to enable 'Create a snapshot immediately and keep the snapshot available to initialize subscriptions' vs. scheduling at a later time

Performance Considerations

Factors that Affect Replication Performance

- Server, storage, network hardware, network, and distance
- Distributor location and isolation
 - Collocated with Publisher, standalone, Subscriber?
 - 1:M Distributor to Publisher? 1:1 Distributor to Publisher?
- Concurrent non-replicated activity (Log Reader penalty)
- Filter complexity
- Very large duration and volume transactions
- Maintenance activities (like re-indexing)
- Table as article in >1 publication
- Agent profile options (some help, some can hurt)
- Using shared vs. independent Agents
- Schedules impact performance (continuous is typically better)
- Key metrics: latency, throughput, concurrency, synchronization timespan, resource overhead
 - We'll cover how to measure later on

Log Reader and Transaction Log Activity

- Log Reader scans transaction log looking for transactions to be replicated
 - This includes non-replicated transactions
- Log Reader latency will be impacted by:
 - Excessive Virtual Log File counts due to many small auto-growths increase the overhead, increasing transactional latency
 - Concurrent non-replicated activity (high volume) will also impact latency
- There are Agent options that can help optimize performance (discussed later)

LogReader.exe Optimizations

- **PollingInterval:**
 - Frequency of the published database transaction log queries for new transactions
 - Default = 5 seconds
 - Decreasing this lowers latency from Publisher to Distributor
 - Tradeoff is the transaction log overhead of the more frequent polling
- **MaxCmdsInTran**
 - Large transactions can cause significant overhead and latency in a replication topology
 - Specifies the maximum number of statements grouped in a transaction to be written by the Log Reader to the distribution database
 - Divides large transactions into smaller transactions
 - Can help with Publisher to Distributor latency
 - Drawback? Transactional atomicity as the Subscriber starts getting rows prior to the end of the original transaction

Hidden Costs of a Single Command Publisher

USE AdventureWorksLT;

UPDATE SalesLT.ProductDescription
SET ModifiedDate = GETDATE();

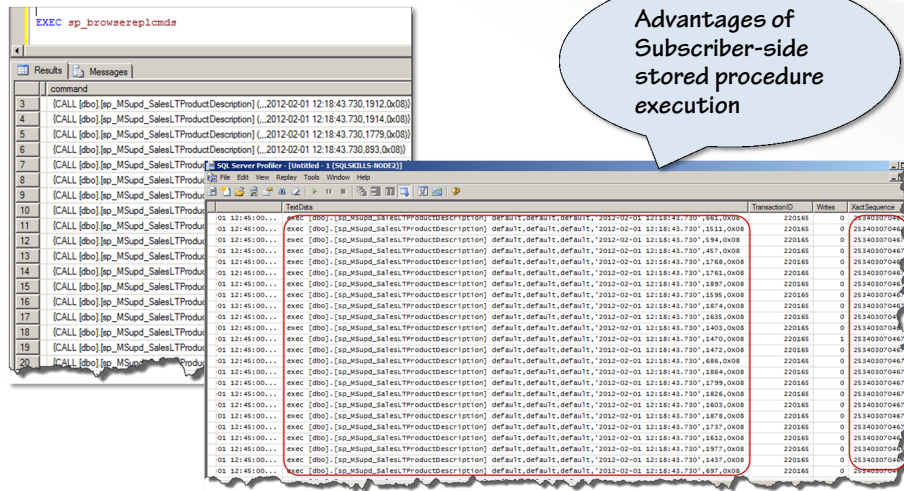
Messages

762 row(s) affected

article_id	partial_command	command	xactid	xact_seqno
1	0	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
2	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
3	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
4	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
5	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
6	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
7	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
8	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
9	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
10	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
11	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
12	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
13	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
14	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
15	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
16	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
17	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
18	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178
19	1	0x7E00000002007B00430041004C004C0020005800640062...	0x00000022000003E90002	0x0000002300000178

Query executed successfully. SQLSKILLS-NODE1 (10.50 SP1) SQLSKILLS-NODE1\Admin... AdventureWorksLT 762 rows

Hidden Costs of a Single Command Distributor and Subscriber



Advantages of Subscriber-side stored procedure execution

TextData	TransactionID	Writes	XactSequence
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070461
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070462
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070463
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070464
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070465
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070466
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070467
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070468
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070469
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070470
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070471
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070472
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070473
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070474
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070475
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070476
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070477
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070478
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070479
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070480
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070481
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070482
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070483
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070484
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070485
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070486
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070487
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070488
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070489
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070490
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070491
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070492
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070493
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070494
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070495
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070496
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070497
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070498
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070499
exec ([dbo].[sp_MSsupd_SalesLTProductDescription] (...))	220365	0	251403070500

Diagnostic Agent Statistics

- Option for Log Reader and Distribution Agent
- -OUTPUT Log Reader Agent parameter allows you to measure fetch time (reading from the transaction log) vs. write time (writing to the distribution database)
 - Both are in milliseconds
 - If writing to a specific log file, make sure the proxy account has permissions to where the file is being written
 - If no path designated, you can see the row details in the Log File Viewer for the job
- Periodic statistics also populated in distribution database's MSlogreader_history and MSdistribution_history
- Set 'HistoryVerboseLevel 2' Agent profile option for added logging
- For any diagnostic options, remove or revert after you're finished troubleshooting (they add to overhead)

Demo

Measuring Log Reader read vs. write time

Agent Execution Frequency

- 'Continuous' execution of the Log Reader and Distribution Agents is typically good for performance
 - You're removing executable start/stop overhead
 - Changes stream continually vs. large 'pile ups' of pending movement

Combining Technologies

Database Mirroring

- **A role can benefit from its database being mirrored if the replication Agents can connect to it after a mirroring failover**
 - Publisher (full support)
 - Agent configuration specifies the failover partner so Agents automatically reconnect to the publication database after a mirroring failover
 - Subscriber (limited support)
 - No support for automatic reconnection of the Distribution Agent
 - Subscriptions must be recreated after mirroring failover of the subscription database
 - Distributor (no support)
 - Server name of the Distributor cannot change, and a mirroring failover keeps the database name but changes the server name
- **Log Reader will wait for log records to harden on the mirror before replicating to the distribution database**
 - You can override this by trace flag 1448 (see KB 937041)

Combining Technologies

Failover Clustering

- **Supported for all replication roles since the virtual network name and instance name failover with the database**
- **Is the only mechanism to provide high availability for the distribution database**
 - Virtualization HA can also be used since the failover occurs at the VM level across hypervisor nodes and the server and instance names remain the same

Monitoring

Setting Warnings and Thresholds

- When the threshold is met or exceeded, a warning is displayed in the status column for the subscription and the publication with which it synchronizes
- You can enable warnings for the following performance conditions:
 - Imminent subscription expiration (All)
 - Exceeding the specified latency (Transactional)
 - Exceeding the specified synchronization time (Merge)
 - Falling short of processing the specified number of rows in a given amount of time (Merge)

Monitoring

Alerts (1)

- Reaching a threshold can also trigger an alert
- Alerts configured at the Distributor (push or pull)
- Typically useful alerts:
 - Replication: Agent failure
 - Replication: Agent retry
 - Replication: expired subscription dropped
 - Replication: Subscriber has failed data validation
 - Replication: subscription reinitialized after validation failure
 - SQL Server performance condition alert on distribution database (file sizes) or a custom monitoring script and job
- Beware of DOS attacks from your own alerts: add delays between firing

Monitoring Alerts (2)

- Don't forget to enable the alerts (some are off by default)
- Alerts may not fire when you expect:
 - E.g. in continuous mode, the 'Agent failure' alert will not raise alerts until Agent retries are done or subscription expiration
 - Default is 2,147,483,647 retries (full number is obscured due to size on the GUI)
 - Use a non-default number of retries based on SLAs
 - If jobs are stopped, no alerting associated with the job
 - One solution is to schedule periodic checks on the agent job status , alerting and/or starting them if stopped

Monitoring with Perfmon Establishing Baselines

- **All Agents**
 - Microsoft SQL Server: Replication Agents: Running
- **Snapshot Agent**
 - SQL Server: Replication Snapshot: Snapshot: Delivered Cmds/sec
 - SQL Server: Replication Snapshot: Snapshot: Delivered Trans/sec
- **Log Reader Agent**
 - SQL Server: Replication Logreader: Logreader: Delivered Cmds/sec
 - SQL Server: Replication Logreader: Logreader: Delivered Trans/sec
 - SQL Server: Replication Logreader: Logreader: Delivery Latency
- **Distribution Agent**
 - SQL Server: Replication Dist.: Dist: Delivered Cmds/sec
 - SQL Server: Replication Dist.: Dist: Delivered Trans/sec
 - SQL Server: Replication Dist.: Dist: Delivery Latency
- **Merge Agent**
 - SQL Server: Replication Merge: Conflicts/sec
 - SQL Server: Replication Merge: Downloaded Changes/sec
 - SQL Server: Replication Merge: Uploaded Changes/sec

Monitoring Using Tracer Tokens

- Transactional replication allows you to measure the latency in a system by inserting a small amount of data known as a tracer token in the transaction log of the publication database
- The trace token is replicated and the replication monitor records how long it takes to arrive at the Distributor and Subscribers
- The token also allows you to identify if data is not reaching the Distributor or subscriber
 - Publisher to Distributor
 - Distributor to Subscriber
- Trace tokens will 'wait in line' behind existing pile-ups

Demo

Monitoring

Troubleshooting (1)

Transactional Replication Tools and General Steps

- Tools include Replication Monitor (including tracer tokens), SQL Server Agent Job history, replication meta-data tables, replication stored procedures, and standard performance methods
- Start by isolating the components within the topology
- **LogReader.exe read issues**
 - Wait statistics and sys.databases log_reuse_wait_desc column
 - Very large transactions or significant amount of throughput
 - Excessive VLFs and auto-growths
 - Transaction log I/O path issues
- **LogReader.exe write issues**
 - Look at wait statistics associated with incoming distribution DB transactions
 - I/O, blocking and other forms of contention in sys.dm_os_waiting_tasks
 - Distribution database size skew and missing cleanup operations
 - Network issues between Publisher and Distributor, protocols used



23

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Troubleshooting (2)

Transactional Replication Tools and General Steps

- **Distrib.exe read issues**
 - Look at wait statistics associated with incoming distribution database transactions
 - I/O, blocking and other forms of contention in sys.dm_os_waiting_tasks
 - Distribution database size skew and missing cleanup operations
 - Network issues between Distributor and Subscriber
- **Distrib.exe write issues**
 - Look at wait statistics associated with incoming distribution database transactions
 - I/O, blocking and other forms of contention in sys.dm_os_waiting_tasks
 - Waits on sp_MSget_repl_commands
 - Concurrent activity at the Subscriber associated with article tables



24

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Demo

Replication Agent wait statistics

Troubleshooting

Data Not Being Delivered to Subscribers

- If the Distribution Agent or Merge Agent is running, use replication validation to perform a binary checksum validation or row count validation against the Publisher and Subscriber(s)
- If the Distribution Agent or Merge Agent is not running, use the tablediff utility to complete the data between the Publisher and Subscriber(s)
- Common causes:
 - Replicated table is filtered
 - One or more Agents are not running or are failing with an error
 - Subscription was initialized without a snapshot and changes have occurred since the publication was created
 - Replication of stored procedure execution produces different results at the Subscriber
 - Data is being deleted by a user, replication script, trigger or another application

Troubleshooting

Publisher and Subscriber Data Does Not Match

- If the Distribution Agent or Merge Agent is running, use replication validation to perform a binary checksum validation or row count validation against the Publisher and Subscriber(s)
- If the Distribution Agent or Merge Agent is not running, use the `tablediff` utility to complete the data between the Publisher and Subscriber(s)
- Check specific rows via `sp_browsereplcmds` and `xact_seqno`
- If tolerable to the application, 'SkipErrors' (see next slide)
- Common causes:
 - Replication of stored procedure execution produces different results at the subscriber
 - Data is being updated by a user, replication script, trigger or another application
 - Constraint violations prevent rows from being inserted, updated, or deleted at the Subscriber



27

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Danger of -SkipErrors

- Distribution Agent parameter where you can designate the skipping of specific errors based on error number so that replication can resume (e.g. - `SkipErrors 2601;2627` to skip constraint violations / duplicates)
- This often leads to data inconsistency between the Publisher and Subscriber that you must either live with or resolve manually
- Impact of this setting may be worse than you expected when it comes to transactional consistency:
 - Imagine a single transaction that makes 300 updates and an error happens at the 150 mark
 - The error is skipped and processes the remaining rows!
- Now imagine if the agent is shared by multiple publications
 - Validate Independent Distribution Agent option
 - 2000: was not independent by default
 - 2005+: default is 'true'
- `sp_setsubscriptionxactseqno` allows the skipping of a specific transaction, but with the same data consistency issues to be aware of



28

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Skipping Specific Commands

- **sp_replshowcmds**
 - Displays transactions pending in transaction log
- **sp_repldone**
 - Allows for skipping commands at Log Reader
 - Large update you may not want propagated (e.g. millions of rows you don't want replicated)
 - Situations where re-initialize would be too time consuming or costly
 - Use as an exception, not as rule
- **sp_replflush**
 - Flushes 'article cache' on transaction log
 - Allows Log Reader to resume work (only one log reading process can be used for a database at a time)

Troubleshooting

Process Could Not Execute 'sp_repldone/sp_replcounters'

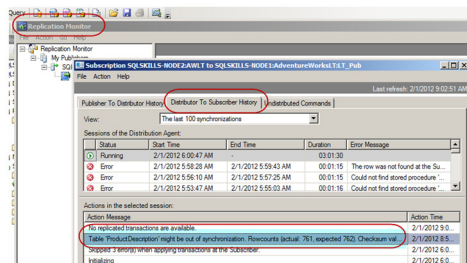
- After restoring the publication database or in the event of corruption (undetected or known), the log reader fails with 'process could not execute 'sp_repldone/sp_replcounters''
- **Issue:**
 - Highest LSN of the Distributor > highest LSN at the Publisher OR
 - Could also appear after using procs that access Log Reader Agent functionality (e.g. sp_repldone) and not followed by sp_replflush
- **Potential fixes:**
 - If encountering the error and there was indeed a recent restore of the database, execute the sp_replrestart procedure at the Publisher
 - Run DBCC CHECKDB to ensure that there are not other issues

Impact of DBCC CHECKDB Repair

- DBCC CHECKDB with REPAIR_ALLOW_DATA_LOSS may affect replicated tables and/or replication metadata tables
 - Repair operations are not recognized by replication executables, which work at the logical layer
 - Data inconsistencies for repairs like removed data pages could cause later agent failures and worse, inaccurate Subscriber application data
- If any repairs involved tables in the distribution database or other replication metadata tables in the user databases, you need to remove and reconfigure the replication topology
- If any repairs involved user tables you can either perform data validation (which may be time consuming and imperfect) or you can reinitialize the subscription
- Covered in more detail by Paul Randal in the SQL Server Pro article, 'Why does using repair invalidate replication subscriptions?'
 - <http://bit.ly/zdIJKn>

Validating Replicated Data

- Launch via SSMS via Local Publications and Local Subscriptions or through the Replication Monitor
- `sp_publication_validation` [1 = rowcount, 2= rowcount + checksum]
- View using Distributor to Subscriber History in Replication Monitor or View Synchronization Status in SSMS
- After validation, the Agent logs success or failure



Error 2812

Could not find stored procedure '%.*ls'

- Distributor to Subscriber fails on message like 'Could not find stored procedure 'dbo.sp_Msupd_SalesLTProductDescription''
- **Step 1: Check for existence of procedure at the Subscriber**
 - May have been dropped by user or process (auditing helps here)
- **Step 2: Execute on the publicationdatabase:**
 - `sp_scriptpublicationcustomprocs [@publication=] 'publication_name'`
- **Step 3: Execute output on subscription database**

SQL Server 2012 Replication Support

- No deprecations, but no significant changes either
- Replicated table/index support for 15,000 partitions
- **Availability Groups:**
 - Transactional, merge and snapshot replication supported
 - Peer-to-peer, bi-directional, reciprocal transactional publications and Oracle publishing are not supported
 - Considerations:
 - Publisher instances need to share a common Distributor
 - Secondary can't be a Publisher
 - Use a remote Distributor NOT on one of the intended replicas
 - New `sp_redirect_publisher` proc used to map publication to AG Listener and distribution database `sp_validate_replica_hosts_as_publishers` to verify all replica hosts can serve as Publishers for the publication (requires read access to replicas)
 - Repl Monitor uses original Publisher name, not AG name

Review

- Performance considerations
- Combining replication with other technologies
- Monitoring
- Troubleshooting

Questions?