

SQLskills Immersion Event

IE3: High Availability & Disaster Recovery

Module 5: Database Snapshots

Paul S. Randal
Paul@SQLskills.com



Overview

- Database snapshot usage
- Database snapshot internals



2

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



Database Snapshots Survey

No - we don't have Enterprise Edition		19%	64
No - no use for them		32%	105
Yes - for reporting/querying off a database mirror		7%	22
Yes - for reporting off a production database		4%	14
Yes - for fast revert in case of admin error		5%	15
Yes - as potential for recovery of deleted data		4%	13
Yes - instead of backups		0%	0
Yes - to reduce locking on production database		1%	3
Yes - in multiple ways		10%	33
Never heard of database snapshots		2%	7
Other? Enter here...		17%	57
Total: 333 responses			

- Source: <http://www.sqlskills.com/BLOGS/PAUL/post/Using-database-snapshots.aspx> (<http://bit.ly/N1Duxl>)

Uses of Database Snapshots

- Isolated historical data for report generation
- Turning a database mirroring server into a reporting server
- Recovery/protection in case of administrative error *
- Recovery/protection in case of user error *
 - Either through data copying or by reverting the entire database
 - Database snapshot has to already exist!
- DBCC CHECKDB (and derivative commands) uses a hidden database snapshot
- NOT a replacement for log backups

How it Works

CREATE DATABASE CreditSS1 (...) AS SNAPSHOT OF Credit

USE Credit;

UPDATE... (pages 4, 9, 10)

Credit: database

Pages



USE CreditSS1;

SELECT... (pages 4, 6, 9, 10, 14)

CreditSS1: database snapshot

Creating a Database Snapshot (1)

Creating a database snapshot

```
CREATE DATABASE [AdventureworksDWSS] ON
(NAME = N'AdventureworksDW_Data',
FILENAME = N'C:\AWDW.mdf_SS')
AS SNAPSHOT OF [AdventureworksDW];
```

- Sparse files are initially allocated as 64KB and then grow as needed

One snapshot file must be specified for each source database data file, including read-only and offline files

Dropping a database snapshot

```
DROP DATABASE [AdventureworksDWSS];
```

Creating a Database Snapshot (2)

- A database snapshot is a transactionally-consistent view of the source database, at the time the database snapshot was created
- Sequence of operations:
 - Checkpoint is performed in source database
 - Sync-point LSN chosen in source database
 - Crash recovery is run INTO the snapshot database FROM the source database
- So although we consider the database snapshot to be empty when it's created, it may have some pages in from the crash recovery that was run

Considerations for Filegroups

- When a source database has one or more offline filegroups, database snapshot creation succeeds with the filegroups offline
- Sparse files are not created for the offline filegroups
- Database snapshot queries on the offline filegroups will fail with I/O errors
- FILESTREAM data is not accessible through a database snapshot, but does not prevent the snapshot being created
 - But does prevent revert from snapshot (see later slide)

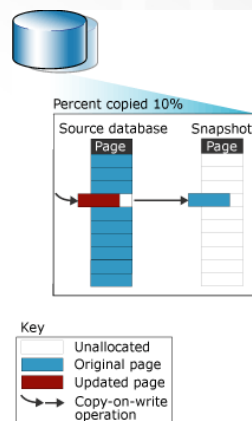
Database Snapshots

Disk Space Requirements

- Extremely space efficient
- Uses a 'copy-on-write' mechanism and NTFS sparse files
 - Really a 'copy-just-before-write' mechanism
 - Managed by the buffer pool
- Pages are pushed into the database snapshot synchronously just before they are changed in the source database
 - A page will only be pushed once
 - Once a page is in the database snapshot, it will not be removed (e.g. if a transaction in the source database rolls back, any pages pushed in the snapshot will stay there)
 - Too many concurrent snapshots can cause performance problems
- Does not create a complete copy of the data
 - Requires extra disk storage only for pages changed in the source database since the database snapshot was created
 - Unchanged pages in the source database are 'shared' with the database snapshot

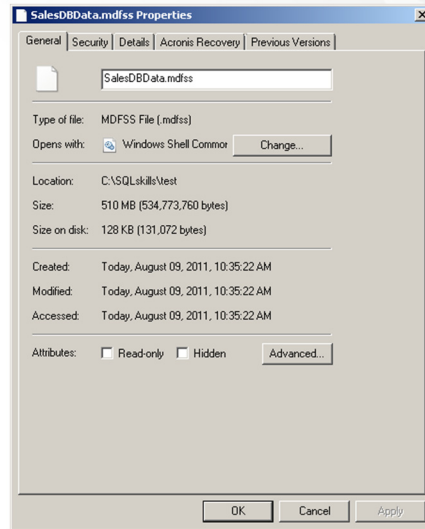
Updating a Database Snapshot

- For every source database file there is a bitmap for the related sparse file
- The bitmap is an in-memory structure
- Updates to page on the source DB (already in memory) are checked against the bitmap to see if the page is already in the sparse file
- If not, then a copy of the page is pushed to the sparse file (copy-on-write)
- For server restarts, the database snapshot still exists but the bitmap will be recreated at the next read attempt made on the database snapshot



How Much Space Used?

- Use Windows Explorer to right click on the NTFS sparse file, select Properties
- Look at the 'Size' and you will see it is the same size of the source database file
- However, look at the 'Size on Disk' and you will see how much space is REALLY used

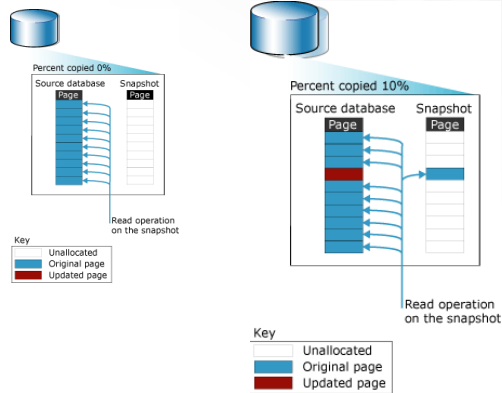


Database Snapshots Memory Requirements

- Although database snapshots are very efficient in terms of disk space, they are NOT efficient in terms of memory usage
- Database snapshots DO share unchanged pages in the source database, but there is a separate copy in memory
- For example:
 - Reading unchanged page X from the source database pulls in a copy of page X from disk into the buffer pool
 - Reading page X from the context of the database snapshot will ALSO pull in a copy of page X from the source database from disk into the buffer pool
 - There will be two in-memory copies of the exact same page image, each taking up 8KB
 - Not optimal, but that's how it works – less complexity in the buffer pool

Reading the Database Snapshot

- When a read of the database snapshot is performed, the bitmap is checked to see where the I/O should go
- No locks taken on reads from the source database through the context of the database snapshot



Database Snapshot Limitations (1)

- Only works on NTFS
- One snapshot file must be specified per data file in the source database
- They cannot be:
 - Written to
 - Refreshed
 - Changed in any way
 - Backed up
 - Source database backups are unaffected
 - Technically this could be done, but not implemented
 - Detached
 - Stored remotely
 - Created for master, model, tempdb

Database Snapshot Limitations (2)

- If the source database goes suspect, so does the snapshot
- All snapshots must be dropped before the source database can be dropped or restored
- Does not support FILESTREAM
- NTFS and Windows limitations may cause problems
 - See PSS blog post: <http://tinyurl.com/4223yoh>
- 3rd-party backup tools may erroneously make files sparse
 - See PSS blog post: <http://bit.ly/flc8PT>
- Beware of 3rd-party file system drivers that do not cope with NTFS sparse files or NTFS alternate streams

Reverting From a Snapshot

- Faster alternative (usually) than restoring from backups
- Reverting pushes all the pages in the database snapshot back into the source database, essentially rewinding it back to the point in time when the database snapshot was created
- All changes since the database snapshot was created are lost
 - No way to apply any log backups to roll forward again
- Reverting breaks the log backup chain
 - Restart using a full or differential backup
- Perform revert operation
 - `RESTORE DATABASE [AdventureWorksDW]`
 - `FROM DATABASE_SNAPSHOT = N'AdventureWorksDWSS';`
- **BEWARE:** reverting from a snapshot resets the transaction log size to 0.5MB → BUG!

Demo

Database snapshots

Review

- Database snapshot internals
- Database snapshot uses and limitations

Resources

- **Whitepaper:**
 - Database Snapshot Performance Considerations under I/O-Intensive Workloads
 - <http://tinyurl.com/44te4ca>
- **Blog posts**
 - <http://www.sqlskills.com/BLOGS/PAUL/category/Database-Snapshots.aspx>



19

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

Questions?

