SQL Server: Myths and Misconceptions

Module 9: Corruption

Paul S. Randal
Paul@SQLskills.com

pluralsight
hardcore developer training

Introduction

- **I always like to tell a room of DBAs that they will all see database corruption at some point in their careers**
- **Dealing with database corruption is way easier if:**
  - You understand how corruption does and does not occur
  - You know how to use DBCC CHECKDB
  - You've practiced recovering from corruption

- **In this module:**
  - Eleven myths around dealing with database corruption

Corruption Myth #1

UNTRUE!!

- **Myth: A SUSPECT database can be repaired by detaching and attaching the database**

- **A SUSPECT database cannot be attached to SQL Server**
- **A SUSPECT database is one where crash recovery could not complete**
- **Attaching a database that was not cleanly shutdown requires completing crash recovery**
  - If crash recovery cannot complete then the database will not attach again
- **Re-attaching a SUSPECT database requires a bit of hacking**
  - Create a dummy database, set it offline, drop in correct files, set it online
- **From SQL Server 2008 onwards, a SUSPECT database cannot be detached using sp_detach_db**

Corruption Myth #2

UNTRUE!!

- **Myth: Running consistency checks on a redundant copy of a database is good enough**

- **Consistency checking a redundant copy of a database does not imply the primary database is free of corruptions**
- **There are two (or more) I/O subsystems involved**
- **None of the SQL Server redundancy technologies ship data file pages between instances**
  - Only transaction log records are shipped to redundant copies
- **I/O subsystem corruption of data files is not propagated**
- **Consistency checks must be run on all copies of a database**

Corruption Myth #3

UNTRUE!!

- **Myth: You can cause corruption using a Transact-SQL statement, or by interrupting a Transact-SQL statement**

- **You cannot cause corruption using Transact-SQL**
- **Interrupting a long-running operation (like a shrink or index rebuild) does not cause corruption either**
  - All changes to a database are logged, and will be rolled back if the operation is cancelled

- **The only time doing something with Transact-SQL could cause database corruption is if there's a bug in SQL Server**
  - This happens occasionally

Corruption Myth #4

UNTRUE!!

- **Myth: Turning on page checksums protects all pages in an upgraded database immediately**

- **Turning on page checksums does nothing**
- **A data file page does not get a page checksum until it is read from disk, changed, and then written back to disk**
  - Restoring from a backup or running DBCC CHECKDB does not do that
- **There is no tool to 'touch' every page and put a page checksum on it**
  - From SQL Server 2008 Enterprise Edition onwards, you could turn Transparent Data Encryption on then off, but that's overkill

- **Also, enabling page checksums does not throw away any torn-page protection in the upgraded database**

Corruption Myth #5

UNTRUE!!

- **Myth: You can avoid running DBCC CHECKDB as long as you use BACKUP … WITH CHECKSUM**

- **A data file page could be corrupt AND have a valid checksum**
- **The data file page could be corrupted in memory by:**
  - A rogue process scribbling in SQL Server's buffer pool
  - A bad memory chip
  - A SQL Server bug
- **And then written to disk with a valid checksum**
- **Only DBCC CHECKDB can detect this**

Corruption Myth #6

RARELY!!

- **Myth: Restarting SQL Server will fix database corruption**

- **Three times in my life I've seen a SQL Server plus I/O subsystem reboot fix corruption because of 'stuck' I/Os**
  - A 'stuck' I/O is one where there's a bug in the software in the I/O subsystem
  - The I/O subsystem returns old/invalid data
- **This is very rare**
- **I do not recommend restarting SQL Server to try to fix corruption**
  - The overwhelming odds are that it will not help
  - It will just add to your downtime

Corruption Myth #7

NOT REALLY!!

- **Myth: Corruptions can disappear**

- **From SQL Server 2005 onwards, if corruptions are reported that means there were corruptions**
  - In SQL Server 2000, DBCC CHECKDB had some bugs that caused phantom corruptions to be reported
- **There is a phenomenon I call 'disappearing corruptions'**
  - If DBCC CHECKDB finds corruptions during regular maintenance but then there's no corruption when DBCC CHECKDB is run later
  - This is because the corrupt pages were deallocated by some other operation that occurred after initial DBCC CHECKDB
  - DBCC CHECKDB only checks allocated pages
- **Beware of 3rd-party file-system filter drivers…**

Corruption Myth #8

UNTRUE!!

- Myth: A clean DBCC CHECKDB guarantees no corruptions

- There is never a guarantee
- A clean DBCC CHECKDB simply means that at the time the data file pages were checked, there were no corruptions
- The I/O subsystem or bad memory could corrupt anything at any time
- This is why regular consistency checks are necessary

Corruption Myth #9

UNTRUE!!

- **Myth: Database repair can fix all corruptions**

- **Repair is not infallible**
- **Repairs are written to be fast, efficient, and 100% correct**
- **Some corruptions cannot be repaired:**
  - PFS allocation pages
  - The boot page
  - File header pages
  - Complex allocation corruptions
  - Some system tables/catalogs
- **These same limitations apply to:**
  - Automatic page repair in database mirroring and AlwaysOn Availability Groups
  - Single-page restore operations

Corruption Myth #10

UNTRUE!!

- **Myth: Repairing system databases/tables is safe**

- **Repair does not work well on system tables/catalogs**
  - Many of these situations are untested
  - Trying to repair certain system tables/catalogs can result in further corruption
- **Master and tempdb cannot be repaired**
  - Repair requires single-user mode
  - Master and tempdb cannot be put into single-user mode
- **Repairing msdb can be dangerous**
  - What if something critical is deleted by repair?
- **Always better to use restore to fix system database/table corruption**

Corruption Myth #11

UNTRUE!!

- **Myth: DBCC CHECKDB causes transactions to roll back**

- **Transactions are not rolled back by DBCC CHECKDB**
  - Even though the error log may print a message implying that they were
- **DBCC CHECKDB creates a database snapshot under the covers**
  - The database snapshot runs crash recovery inside itself
  - This creates a transactionally-consist view of the database being consistency checked
- **Unfortunately, the message in the error log does not make that clear**

- **Also, DBCC CHECKDB does not get run automatically when the instance starts**
  - It's just reporting the last-known-good time in the error log