

SQLskills Immersion Event

IE3: High Availability & Disaster Recovery

Module 10: Corruption Detection and Recovery

Paul S. Randal
Paul@SQLskills.com



Why Is This Module Important?

- **Corruption does happen, mostly caused by IO subsystem problems**
- **People don't realize they have corruption until too late**
 - Either they don't know how to check for corruption or they miss the warning signs
- **People don't know what to do when they do have corruption, leading to:**
 - More data loss and downtime than necessary
 - Monetary and even job losses
 - Overall lowered perception of SQL Server
 - Makes it harder to convince management that SQL Server is Enterprise capable



What Can Happen to an Unprepared DBA Confronted by Corruption?



Practice Makes Perfect

- **If you're recovering from a disaster, or helping a client recover from a disaster:**
 - You must know what you're doing
 - You must have practiced
 - You must NOT make things worse
- **Most of you have restored simple backups or run DBCC CHECKDB, but there are some things that most people have not done**
 - These are the things that will set you apart from others
 - Your colleagues and clients will love you if you can save them while minimizing downtime and data loss

Overview

- I/O errors and what they mean
- Page protection options
- DBCC CHECKDB
 - What it does
 - Best practices and FAQ
 - Interpreting CHECKDB output
 - How repair works
- Recovering from corruption
- Horror stories from the wild

How Does Corruption Occur?

- It's the I/O subsystem, usually always
 - 1 out of 10,000 corruptions are bad memory
 - 1 out of 10,000 corruptions are SQL Server bugs
- Consider the MTBF of disks, you're going to have a problem at some point in your career
- Jim Gray likened a disk head in a 15000rpm disk to a 747 flying at 500mph about ¼ inch above the ground
 - What happens in a crash?
- Corruptions cannot be caused by:
 - Anything an application can do
 - Interrupting a shrink, rebuild, or long-running batch
 - Propagation by log-shipping, replication, mirroring

I/O Errors

- **Three types**
 - 823: a hard I/O error
 - 824: a soft I/O error
 - 825: a read-retry error
- **Nice error messages in 2005+**

Msg 824, Level 24, State 2, Line 1

SQL Server detected a logical consistency-based I/O error: incorrect checksum (expected: 0x7232c940; actual: 0x720e4940). It occurred during a read of page (1:143) in database ID 8 at offset 0x0000000011e000 in file 'c:\sqlskills\broken.mdf'. Additional messages in the SQL Server error log or system event log may provide more detail. This is a severe error condition that threatens database integrity and must be corrected immediately. Complete a full database consistency check (DBCC CHECKDB). This error can be caused by many factors; for more information, see SQL Server Books Online.

- **Logged in msdb.dbo.suspect_pages**
 - Input into single-page restore operations

Read-Retry

- **Present in SQL Server 2000 for limited circumstances**
- **Extended in SQL Server 2005 for data pages**
- **Reads that fail because of I/O errors are retried 4 times**
- **Eventual failure results in an I/O error**
- **Eventual 'success' results in an error in the error log:**
 - A read of the file <<FILE NAME>> at offset <<PHYSICAL OFFSET>> succeeded after failing <<RETRY COUNT>> time(s) with error: <<DETAILED ERROR INFORMATION>>. Additional messages in the SQL Server error log and system event log may provide more detail. This error condition threatens database integrity and must be corrected. Complete a full database consistency check (DBCC CHECKDB). This error can be caused by many factors; for more information, see SQL Server Books Online.
- **Early warning of impending doom as read-retry means the I/O subsystem is returning incorrect data to SQL Server**
- **KB article: <http://support.microsoft.com/kb/2015757>**

SQLIOSim

- Use before even installing SQL Server to stress test the I/O subsystem underneath SQL Server
- Freely available for download since October 2006
 - x86, x64, IA64 versions
- Replaces SQLIOStress from SQL Server 2000
- Simulates I/O workload of SQL Server
 - Configurable workloads (e.g., bulk load, DBCC, read-ahead)
- Not just testing the hardware...
- See blog post:
 - <http://www.sqlskills.com/blogs/paul/post/How-to-tell-if-the-IO-subsystem-is-causing-corruptions.aspx> (<http://bit.ly/g4qzwp>)
- Interpretation:
 - <http://blogs.msdn.com/jimmymay/archive/2009/09/27/sqliosim-parser-by-jens-suessmeyer-yours-truly.aspx> (<http://bit.ly/RRHVAd>)

Page Protection Options

- SQL Server allows pages to be 'protected' on disk from corruptions
- Allows fast detection of corruptions
- Set using
 - ALTER DATABASE SET PAGE_VERIFY <option>
- Three options:
 - NONE (don't do this...)
 - TORN_PAGE_DETECTION
 - CHECKSUM

Torn-Page Detection

- An 8k page is really 16 x 512-byte disk sectors
- Possible for a page to be partially written in the event of a power failure, i.e. a torn page
- Torn-page detection protects SQL Server against this
 - Takes two-bits from each sector, stores them in the page header and writes an alternating bit pattern in each sector
 - On subsequent read, if the pattern is disrupted, the page is torn
- Does not detect corruptions within a disk sector

Page Checksums (1)

- Per-page checksum
 - Written as the very last thing SQL Server does on a physical write
 - Checked as the very first thing SQL Server does on a physical read
- Provides the 'smoking gun' that the error is not due to SQL Server
- On by default for new databases in SQL Server 2005+
 - Added to tempdb for 2008+
- Checksum failures result in an 824 error
- Negligible CPU overhead
- Error-detecting, not error correcting
- Superset of torn-page detection

Page Checksums (2)

- **Checked when:**
 - Page is read normally
 - Page is read during CHECKDB
 - Page is read during BACKUP WITH CHECKSUM
 - Page is contained within a checksum'd backup
- **Switching it on doesn't do anything until pages are written...**
- **Blog post:**
 - <http://www.sqlskills.com/blogs/paul/post/How-to-tell-if-the-IO-subsystem-is-causing-corruptions.aspx> (<http://bit.ly/g4qzwp>)

Memory Corruption

- **The buffer pool periodically validates page checksums of clean pages in the buffer pool**
 - Failures results in an 832 error (do not confuse with 823)
- **SQL Server 2012 on Windows Server 2012 also supports automatic correction of some memory corruptions of buffer pool memory**
 - Corrupt clean pages are re-read from disk
 - Corrupt dirty pages cannot be repaired
 - See Glenn's post at <http://bit.ly/SmZLLA> for more details

Automatic Page Repair

- SQL Server 2008+ tie in the detection of a corrupt page to database mirroring so that corrupt pages can possibly be automatically repaired
- Works for 824 'soft-I/O' errors, some 823 'hard-I/O' errors, 829 'in restore' errors
 - Errors are hit by queries reading the page or by DBCC CHECKDB
- Corrupt pages on the principal AND mirror can be repaired
- Repairs are asynchronous, corrupt pages are unusable until fixed
- NOTE: Only meant to be a band-aid to prevent downtime. It is NOT a substitute for having alerts for high-severity errors and taking affirmative action to rectify/prevent them.
- More information in the database mirroring module

DBCC CHECKDB

- The only way to read all allocated pages in the database
 - Use to force page checksums to be checked
- Choose between full checks and WITH PHYSICAL_ONLY
- Many algorithms to minimize runtime and run ONLINE since SQL Server 2000
 - Compared with 6.5 or 7.0
- New features in 2005+
 - Progress reporting, data purity, indexed views, last known good, no false failures...
- New features in 2008+
 - Long-running checks moved under WITH EXTENDED_LOGICAL_CHECKS
- Blog post series:
 - <http://www.sqlskills.com/blogs/paul/category/CHECKDB-From-Every-Angle.aspx> (<http://bit.ly/TFvxmG>)

How To Run DBCC CHECKDB

- **By default, CHECKDB will:**
 - Only return the first 200 errors (fixed in latest versions)
 - Return lots of info that's distracting in a corruption situation
- **Use the following command with only these options:**
 - DBCC CHECKDB (yourdb) WITH ALL_ERRORMSG, NO_INFOMSGS
- **If it's taking longer than usual, that usually means that it found some corruption**
 - Check the error log for message 5268 from SQL Server 2005 SP2+ to see if it's rescanning some data
- **Most importantly, wait for it to complete!**

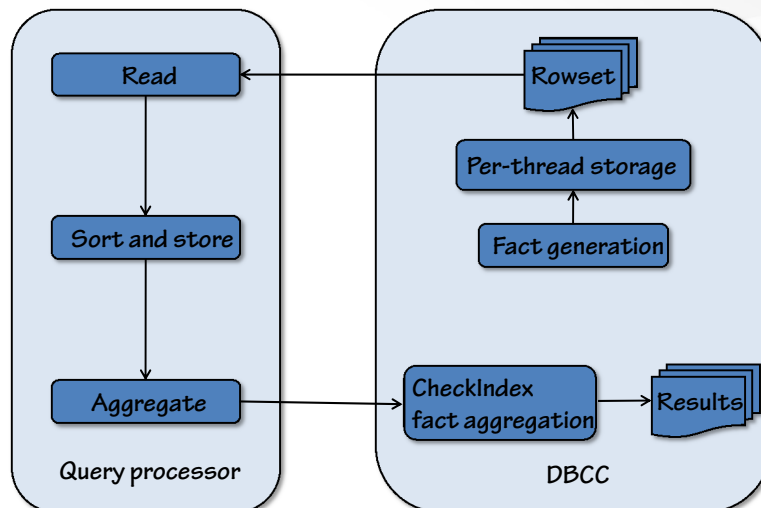
What Exactly Does CHECKDB Do? (1)

- **Primitive checks of critical system tables**
 - Problems? Game over.
- **Allocation checks**
- **Logical checks of critical system tables**
- **Logical checks of all other tables**
 - Including nonclustered indexes, LOB, FILESTREAM...
- **Service Broker data validation**
- **Metadata checks**
- **Indexed view, XML index, spatial index checks**
 - Only under the WITH EXTENDED_LOGICAL_CHECKS option in SQL Server 2008+
- **Repairs are done at each stage (if specified and possible)**
- **What about when WITH PHYSICAL_ONLY is used?**

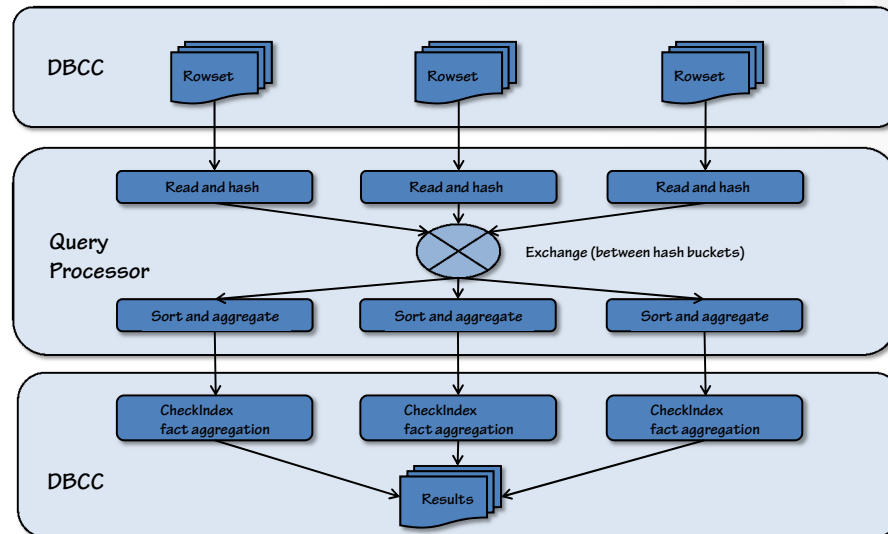
What Exactly Does CHECKDB Do? (2)

- Another way to think of it is that CHECKDB does:
 - Primitive system table checks, then
 - DBCC CHECKALLOC, then
 - DBCC CHECKTABLE of system tables, then
 - DBCC CHECKTABLE of user tables, then
 - DBCC CHECKCATALOG
- It does not do:
 - DBCC CHECKIDENT
 - DBCC CHECKCONSTRAINTS

What Exactly Does CHECKDB DO? (3)



What Exactly Does CHECKDB DO? (4)



Other CHECK* Commands

- **DBCC CHECKALLOC**
 - Performs allocation checks on a whole database
- **DBCC CHECKTABLE**
 - Performs logical checks on a single table
- **DBCC CHECKFILEGROUP**
 - Performs database allocation checks plus logical checks on all tables within a specified filegroup
- **DBCC CHECKCATALOG**
 - Performs cross-consistency checks between metadata
- **DBCC CHECKIDENT**
 - Verifies/resets table identity values
- **DBCC CHECKCONSTRAINTS**
 - Verifies constraints

How Are CHECK Commands Online?

- **Very first thing the command does is create a transactionally-consistent, point-in-time view of the database**
 - If the database is not already a snapshot, read-only, or single-user
- **This is done using a hidden database snapshot**
 - Created as alternate-streams of the existing data files
 - In 2000 it used transaction log analysis, which was slooow
- **Sometimes problems occur:**
 - The snapshot runs out of space
 - No permissions to create the snapshot
- **Solution: create your own snapshot and check that**

Things to Be Aware Of

- **Fact storage uses TEMPDB, so TEMPDB must be sized appropriately**
 - Use WITH ESTIMATEONLY
- **Internal database snapshots use NTFS alternate streams on existing database files**
 - Some 3rd party software use kernel filter drivers that do not cope with these and so CHECKDB thinks the snapshot is corrupt
- **Beware of sudden very-long run-times**
 - Most likely a corruption has triggered a deep-dive algorithm

How Often Should I Run CHECKDB?

- **It all depends on a combination of:**
 - ❑ Stability of I/O subsystem
 - ❑ Backup strategy
 - ❑ Acceptable downtime if corruption occurs
 - ❑ Acceptable data-loss if corruption occurs
 - ❑ Window available to take the extra I/O and CPU load
 - ❑ What kind of system it is (e.g., production, test, backup)
 - ❑ How the data is partitioned
- **Examples:**
 - ❑ No backups and persistent corruptions
 - ❑ VLDB with very low downtime/data-loss tolerance

How Long Will CHECKDB Take to Run?

- **Depends on many factors:**
 - ❑ Size of the database
 - ❑ Concurrent I/O load on the server
 - ❑ Concurrent CPU activity on the server
 - ❑ Concurrent update activity on the database
 - ❑ Throughput capabilities of the I/O subsystem
 - ❑ Number of CPUs on the server
 - ❑ Speed of the disks where TEMPDB is placed
 - ❑ Complexity of the database schema
 - ❑ Which options were specified
 - ❑ Number and type of corruptions that exist
- **See blog post for more details:**
 - ❑ <http://www.sqlskills.com/blogs/paul/post/CHECKDB-From-Every-Angle-How-long-will-CHECKDB-take-to-run.aspx> (<http://bit.ly/RRL570>)

Making CHECKDB Go Faster...

- <http://support.microsoft.com/kb/2634571>
- Recent work has been done to improve the performance of CHECKDB by:
 - Allowing it to check everything as a single batch
 - Slightly changing the I/O pattern it generates
- Ported to 2008, 2008R2, 2012
 - Check the KB article to see whether trace flags are required
- Be aware this will push the I/O subsystem much harder
- Bob Ward's write-up on this:
 - <http://bit.ly/yAFY7W>

How To Consistency-Check VLDBs?

- DBCC CHECKDB of a multi-TB database can take hours!
- Five options for reducing the run time:
 - Don't run any checks
 - Run WITH PHYSICAL_ONLY
 - Break up the checks
 - Use partitioning
 - Use another system
- Don't give up—with a little thought it's possible
- See blog post for more details:
 - <http://www.sqlskills.com/blogs/paul/post/CHECKDB-From-Every-Angle-Consistency-Checking-Options-for-a-VLDB.aspx> (<http://bit.ly/TFw8om>)

Consistency Checking Survey (1)

What method do you use to run consistency checks on your production database (s)?

Run DBCC CHECKDB with no options on the production database		62%	261
Run DBCC CHECKDB WITH PHYSICAL_ONLY on the production database		10%	42
Run DBCC CHECKDB on a restored backup on another server		11%	47
Run DBCC CHECKDB on a database snapshot on a mirror database		1%	4
Spread consistency checking over several days using DBCC CHECKTABLE		0%	1
Spread consistency checking over several days using DBCC CHECKFILEGROUP		1%	4
Use BACKUP WITH CHECKSUM to validate page checksums, no DBCCs		1%	4
Run DBCC CHECKDB on a log shipping secondary, or 2012 AG secondary		1%	3
Run DBCC CHECKDB on a SAN copy/mirror		2%	7
Create your own database snapshot of the production database and run DBCC CHECKDB on it		0%	2
Other? Enter here...		11%	47
Total: 422 responses			

- Source: <http://www.sqlskills.com/BLOGS/PAUL/post/Importance-of-how-you-run-consistency-checks.aspx> (<http://bit.ly/Hq3m77>)



29

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

Consistency Checking Survey (2)

How often do you run consistency checks? (regardless of "how" you run them)

What are consistency checks?		9%	25
Never		5%	14
Only when corruption is detected some other way		8%	22
Only during an event like an upgrade or migration		1%	2
Regularly, but less than monthly		3%	8
Monthly		5%	14
Weekly		37%	103
Daily		25%	69
More frequently than daily		1%	3
Only after performing a restore, or after a failover		1%	3
Other? Enter here...		5%	13
Total: 276 responses			

- Source: <http://www.sqlskills.com/BLOGS/PAUL/post/Importance-of-running-regular-consistency-checks.aspx> (<http://bit.ly/1mUuRw>)



30

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

Did Something Go Wrong?

- So you setup a regular job to run consistency checks, and turned on page checksums – how can you tell if it goes wrong?
- There has to be some kind of monitoring otherwise you'll never know!
- Manual monitoring is time-consuming and prone to being forgotten
- **Solution: Agent alerts, or other monitoring (e.g. SCOM)**
- **Create alerts for:**
 - Severity 19 errors and above
 - Any user-defined errors (e.g. flagging that CHECKDB job failed)
 - Anything else you're interested in
- See <http://www.sqlskills.com/BLOGS/PAUL/post/Easy-monitoring-of-high-severity-errors-create-Agent-alerts.aspx> (<http://bit.ly/hrBiTK>)
- And <http://spaghettidba.com/2011/11/28/email-alert-dbcc-checkdb/> (<http://bit.ly/TFwKdK>)

First Hints of Corruption...

- Application/user connections get broken
- Users report 823 or 824 errors
- Backup jobs start failing
 - Error 3043 – backup detected checksum errors
- Agent alerts start firing
- Maintenance jobs start failing
- Errors in the SQL Server error log
- All these hint that you've got corruption

As Soon As Corruption Is Suspected

- **No need to panic!**
 - Panic leads to mistakes and downtime/data-loss
- **Go to the DR run-book**
- **Determine the extent of the corruption**
 - Run DBCC CHECKDB
 - Look in the SQL Server error log, Windows event log
 - Check maintenance job history
- **Check what backups are available**
- **Wait for CHECKDB to finish before doing anything else**
 - You may not NEED to do anything intrusive/destructive

Interpreting CHECKDB Output (1)

- So, CHECKDB completes and you have a bunch of cryptic error messages. Now what?
- There are over 100 errors that CHECKDB can output, some with over 200 states
- Figuring out what one error means isn't too bad
 - MSDN has some of them published and is working to publish the rest over the next year
- Figuring out multiple errors is very hard and usually isn't worth the time
- There are some tips and tricks you can use...

Interpreting CHECKDB Output (2)

- **Did CHECKDB fail?**
 - If it stops before completing successfully, something bad has happened that is preventing CHECKDB from running
 - This means there is no choice but to restore from a backup as CHECKDB cannot be forced to run (and hence repair)
- **Examples of fatal (to CHECKDB) errors**
 - 7984 – 7988: corruption in critical system tables
 - 8967: invalid states within CHECKDB itself
 - 8930: corrupt metadata in the database such that CHECKDB could not run
 - See 'Understanding DBCC Error Messages' in the BOL for DBCC CHECKDB for more details

Demo

Example fatal errors to CHECKDB

Interpreting CHECKDB Output (3)

- **Are the corruptions only in nonclustered indexes?**
 - If recommended repair level is REPAIR_REBUILD, YES!
 - Otherwise, check all the index IDs in the errors – if they're all greater than 1, then YES!
- **If YES, you could just rebuild the corrupt indexes**
 - Depends on the error, and the size of the index
 - But, what caused the corruption?
 - If you just rebuild, the corruption will probably happen again (especially if caused by the I/O subsystem)
 - Rebuild by disabling the index and then rebuilding, all wrapped in a transaction to prevent constraint violations
 - Make sure you do root-cause analysis and take preventative measures

Demo

Nonclustered index corruption

Interpreting CHECKDB Output (4)

- **Was there an un-repairable error found?**
 - 8909, 8938, 8939 (page header corruption) errors where type is 'PFS'
 - 2570 error: invalid data for the column type
 - 8992 error: CHECKCATALOG (metadata mismatch) error
 - Plus a few more obscure ones
 - E.g. an 8904 error (extent is allocated to two objects). This is usually repairable except in the case where the extent is marked as mixed and dedicated, and has pages allocated to multiple objects. The repair is too complicated and/or destructive so is not attempted.
- **None of these can be automatically repaired**
 - But if you don't have a backup without these corruptions, you may be able to fix the 2570 and 8992 errors...
 - <http://support.microsoft.com/kb/923247> for 2570 errors

Demo

Fixing data purity errors

Recovering Using Backups

- **Best way to avoid data loss**
- **Not necessarily the best way to avoid downtime**
 - Depends what kind of backups are available
 - Although backup compression in SQL Server 2008+ helps...
- **Plethora of options available**
 - Full database backup is a good starting point
 - Series of transaction log backups as well is much better
- **Remember:**
 - Backups have to exist to be useful
 - Backups have to be valid to avoid data loss

What if SQL Server is Unavailable?

- **You always want to try a tail-of-the-log backup**
- **If the SQL Server instance is unavailable, have to perform a 'hack-attach' of the log file to another instance to perform the backup**
- **Steps to perform:**
 - Create dummy database with same name
 - Set database offline
 - Delete all data and log files
 - Drop in salvaged log file you want to back up
 - Try to bring database online
 - Perform tail-of-the-log backup
- **Note: only works on the same version of SQL Server**

Demo

Last resort tail-of-the-log backup

Single-Page Restore

- **One or more pages can be restored**
 - Incredibly valuable if database not architected for partial database availability
- **Backups must exist to allow the page to be restored to same point in time as PRIMARY filegroup**
- **Not all page types can be single-page restored**
 - Boot page
 - Fileheader pages
 - Allocation bitmaps (not including IAM pages)
 - Certain pages in hidden, critical system catalogs

Demo

Using single-page restore

Looking Into Log Backups

- **fn_dblog**
 - select * from fn_dblog (startLSN, endLSN)
 - Used to dump the contents of the transaction log
- **fn_dump_dblog**
 - select * from fn_dump_dblog (startLSN, endLSN, 'DISK'|'TAPE', devicenum, backuppath, DEFAULT, DEFAULT,...)
 - Used to dump the contents of the transaction log in a backup
- You can make inching through the log a bit easier by trying to find the point to restore to

Demo

Looking into log backups

Restore vs. Repair (1)

- Multiple decision points that could short-circuit the decision process
- Do you still have a database?
 - No – you must restore from a backup
- Do you have working backups?
 - No – you must use repair, or restore a damaged backup with `CONTINUE_AFTER_ERROR`, or extract data to a new database
- Is the log damaged?
 - Yes – you must restore, or run emergency mode repair, or extract to a new database

Restore vs. Repair (2)

- **Did CHECKDB fail?**
 - Yes – you must restore or extract
- **Is it just nonclustered indexes that are damaged?**
 - Yes – maybe rebuild them manually
- **Are there any un-repairable errors?**
 - Yes – you must restore or extract
- **If you're still able to make a repair/restore choice:**
 - Consider your down-time and data-loss Service Level Agreements
 - Use whichever option you can which allows you to limit down-time and data-loss while still staying within the SLAs

Running Repair

- **Not always a last resort compared to restore**
 - Depends what backups are available
 - Depends if downtime more important than data loss
- **Remember repair requires SINGLE_USER mode**
- **Try to repair smallest thing**
 - CHECKDB is largest, then
 - CHECKTABLE, then
 - CHECKALLOC
 - Undocumented REPAIR_ALL option to fix GAM, SGAM pages
- **Note: most metadata tables can't be repaired**
- **Note: repair is untested on many system tables**
- **Be careful about running repair on msdb**

How Does Repair Work?

- **What is the purpose of repair?**
 - Make the database structurally consistent
 - Tries to be as fast as possible
- **How does it know what to repair?**
 - From the list of corruptions it just found
- **How does it choose what to repair first?**
 - Runs most intrusive errors first
- **Did it repair everything?**
 - Check the output – count of errors found and fixed
 - Be careful – some corruptions could be masked by others
- **Why aren't repairs online?**
 - Hard enough to get them right *offline*...

Beware of REPAIR_ALLOW_DATA_LOSS

- **Repair fixes structural inconsistencies by de-allocating**
 - (Not REPAIR_REBUILD, but indexes should be fixed manually)
 - This is the fastest and most provably correct way
- **Repair doesn't take into account:**
 - Foreign-key constraints
 - Inherent business logic and data relationships
 - Replication (see BOL for DBCC CHECKDB)
- **Before running repair, protect yourself**
 - Take a backup and quiesce replication topologies involved
- **After running repair, check the data**
 - Consider running DBCC CHECKCONSTRAINTS
 - Fix up any replication topologies involved

Examples of Repairs

- **What does repair do to fix a:**
 - Missing nonclustered index row
 - Just insert the missing record
 - Corrupt data record
 - Maybe delete the record, maybe the whole page
 - Extent allocated to multiple objects
 - Deeper examination of the pages in the extent
- **Remember there are some un-repairable errors**
 - System table clustered index data pages
 - PFS pages
 - Data purity errors

Demo

Using repair

Misconceptions Around Repair

- Repair will not cause data loss (it depends)
- Repair should be run as the default (no)
- You can run repair without running DBCC CHECKDB (no)
- As soon as you've run repair, continue as normal (no)
- Repair can always fix everything (no)
- Repair is safe on system databases (no)
- You can repairs online (no)
- REPAIR_REBUILD will fix everything (no)
- Repair fixes up constraints (no)
- EMERGENCY mode repair will always work (no)
- You can undo repairs (it depends)
- Repairs are propagated to replication subscribers (no)

System Table Corruption

- **Hidden system tables cannot be accessed unless the Dedicated Admin Connection is used**
 - Sqlcmd -A or prefix connection string with admin:
 - TF7806 at startup allows DAC connections under all circumstances
- **These tables cannot be changed unless the server is booted with -m**
- **From 2008 onwards, the fact that a system table was directly modified is noted in the database metadata, just like log rebuilds**
 - Makes your database technically unsupported
 - DBCC TRACEON (3604)
 - DBCC DBTABLE
 - Look at dbi_updSysCatalog and dbi_RebuildLogs

Rebuilding System Table Indexes

- Structurally corrupt system table clustered indexes cannot be repaired
- Structurally corrupt system table nonclustered indexes usually can be rebuilt by repair
 - May have to play around with CHECKTABLE to find which table is corrupted as CHECKDB may fail
 - Not guaranteed to work... be careful
- I would try to avoid running repair on system tables completely

Demo

Fixing metadata corruption

Damaged/Missing Log with Clean Shutdown

- The transaction log is essential for crash recovery to work
- However, a damaged or missing transaction log does not matter if the database was cleanly shut down
 - The transaction log is not required as there is no crash recovery to run
- In that case, a new transaction log can be created
 - Requires re-attaching the database using CREATE DATABASE and using either the FOR ATTACH or FOR ATTACH_REBUILD_LOG options
 - FOR ATTACH creates a single 0.5MB transaction log file, but only works if there was previously one transaction log file
 - FOR ATTACH_REBUILD_LOG also creates a single 0.5MB transaction log file, but works no matter how many log files there were previously
- Make sure to provision the correct size and auto-growth afterwards

Damaged/Missing Log Without Clean Shutdown

- If the database was not cleanly shut down, crash recovery must be performed before the database can be brought online
- If the transaction log is damaged or missing, the database will be in one of two states:
 - RECOVERY_PENDING: crash recovery has to run but could not start
 - SUSPECT: crash recovery started but could not complete
- Recovering from this situation requires restoring from backups, or using EMERGENCY mode to access the database

EMERGENCY Mode

- **EMERGENCY mode is also known as 'bypass recovery' mode, as this is exactly what it instructs the Storage Engine to do**
 - ALTER DATABASE mydb SET EMERGENCY;
- **Although this allows access to the database, the data is transactionally inconsistent**
 - Furthermore, the database itself may be structurally inconsistent
- **You can try to extract data out into a new database**
- **Without backups, the only way to recover the existing database is to try EMERGENCY-mode repair**

EMERGENCY-Mode Repair

- **I added this to SQL Server 2005 as a documented and supported way to attempt recovery with a damaged transaction log**
 - It used to be possible using DBCC REBUILD_LOG in SQL Server 2000
- **The database must be in EMERGENCY and SINGLE_USER modes**
- **Running DBCC CHECKDB (mydb, REPAIR_ALLOW_DATA_LOSS) does the following:**
 - Run as much crash recovery as possible, skipping damaged log records
 - Build a new transaction log file
 - Run a full DBCC CHECKDB with REPAIR_ALLOW_DATA_LOSS
 - Bring the database online
- **Note: this not 100% guaranteed to work in all scenarios**

Damaged/Missing Log of Attached Database

- If the SQL Server runtime encounters transaction log corruption, for instance while rolling back a transaction, the database will be set offline with a status of SUSPECT
- Recover from this using backups or EMERGENCY mode
- If there are no active transactions that could encounter the corruption, DBCC CHECKDB or a transaction log backup may fail
- To recover from this:
 - Switch the database to the SIMPLE recovery model
 - Issue a CHECKPOINT to force the transaction log to clear
 - Switch back to the original recovery model
- This should hopefully clear the damaged log records
- Note: this breaks the log backup chain

Things That People Often Try FIRST

- Restart SQL Server
 - Just wastes time and delays getting back online
 - Although I've seen it work rarely when the I/O subsystem is doing stale reads
- Immediately jump to a last resort and cause data loss without working through options
 - Running repair
 - Rebuilding the transaction log
- Detach a suspect database
 - It will fail to attach again: now the situation is even worse!
 - This is the 2nd worst state to be in
 - However, there's a trick you can use...

Demo

Using EMERGENCY mode

What If You Don't Have a Database At All *OR* Any Kind of Backup?

- Total data loss - *this* is the worst state to be in
- You might have no choice apart from manual re-entry, or



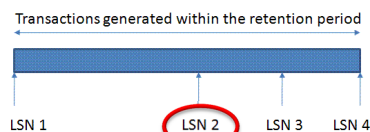
- URLT
 - Update Resume, Leave Town ☺

Recovering After a Failover

- **Nothing specific to do for:**
 - Failover clustering
 - Database mirroring
 - Log shipping
- **What about replication and mirroring combined?**
 - Publisher works fine
 - Subscriber works in 2008 but needs some setup:
 - Minimum retention period in distributor must be set
 - Initialize-from-LSN initialization must be done of new subscription after failover

Transactions in the Distribution Database

- This a view into the distribution database after a failover



- **Each transaction has a sequence number (LSN)**
 - LSN 1: oldest transaction still in the distribution database
 - LSN 2: last transaction applied to the Subscriber mirror
 - LSN 3: last transaction applied to the Subscriber principal
 - LSN 4: most recent transaction entered into the distribution database by the Log Reader Agent
- (LSN 3 minus LSN 2) is how far behind Subscriber mirror was, and all transactions needed to bring
- Subscriber mirror up-to-date after a failover

The Ultimate Advanced Recovery...

- **Recreating database pages using a hex editor**
 - E.g. file header page of file ID = 1
- **This can be done, but it's very, very tricky**
- **Fixing pages using a hex editor is easier**
- **But still pretty tricky – I rarely do it**
 - I last did this for a paying client in August 2011 to fix broken off-row LOB links in sysindexes in a 2000 production database
 - No – I'm not going to demo this live ☺
 - It took several hours to get it right last time I tried

Other Things To Consider

- **Pulling data from database snapshot**
 - Doesn't support FILESTREAM data
- **Pulling data from log shipping secondary**
- **Recovering data from older backup by matching with page IDs/keys from DBCC CHECKDB pre-repair**

Customer Examples to Learn From

- All of the following examples happened while I was at Microsoft
- I was involved with each customer
- No names will be divulged
- In increasing magnitude of loss
- These are just the worst ones...

Examples to Learn From #1

- **Story**
 - Database is corrupt and takes literally *days* to bring online
 - Customer is a small bank in the US
- **Cause**
 - Customer only had a full backup from January 2004 plus log backups every ½ hour until April 2004 when corruption hit
- **Resolution**
 - Restored ALL the backups (approx 5000!) – luckily none were corrupt
- **Cost**
 - Aggravation to bank customers while bank was down for several days
- **Lesson Learned**
 - Plan a backup strategy that allows an efficient and quick restore

Examples to Learn From #2

- **Story**
 - Database lost with corrupt backup
 - Customer is a large investment firm and the database stores all 401k data for all its customers
- **Cause**
 - Database was backed up prior to moving to new hardware
 - Old h/w flattened *before* database restored on new h/w
 - Backup corrupt
- **Resolution**
 - Force corrupt backup to restore and patch together broken data
- **Cost**
 - Job losses (from DBA to VP level)
- **Lessons Learned**
 - Always validate backups
 - Always restore and check *then* delete

Examples to Learn From #3

- **Story**
 - Database and application down for 3 days after corruption
 - Database stores stock trades for UK customers of a large US bank
- **Cause**
 - Started DBCC CHECKDB but took too long so killed it
 - Started to restore from backups but found they were corrupt
 - Re-ran DBCC CHECKDB again and let it complete
- **Resolution**
 - Eventually found it was a nonclustered index corruption that could have been fixed with an index rebuild
- **Cost**
 - Customer fined >\$10m by UK government
- **Lessons Learned**
 - Know how long DBCC CHECKDB takes, and it may take longer with corruption
 - Validate backups

Examples to Learn From #4

- **Story**
 - DBA comes to work to find the database and all backups gone
 - Database stores all tax records for a US state
- **Cause**
 - 3rd party technician accidentally formatted the drive where the database *AND* backups (!!!) were stored while installing hardware
- **Resolution**
 - 200 staff members spent 3 weeks entering data for all state residents from scratch
- **Cost**
 - Unsubstantiated report of cost to the state of more than \$2 *billion* in lost tax revenue
 - No job losses...
- **Lesson Learned**
 - Always have multiple backups, including off-site
 - Carefully guard access to infrastructure

Review

- I/O errors and what they mean
- Page protection options
- DBCC CHECKDB
 - What it does
 - Best practices and FAQ
 - Interpreting CHECKDB output
 - How repair works
- Recovering from corruption
- Horror-stories from the wild

Blog Posts (1)

- **Conference corruption demo scripts and example corrupt databases**
 - <http://www.sqlskills.com/BLOGS/PAUL/post/Conference-corruption-demo-scripts-and-example-corrupt-databases.aspx> (<http://bit.ly/1UC4N>)
- **Everything you ever wanted to know about CHECKDB**
 - <http://www.sqlskills.com/blogs/paul/category/CHECKDB-From-Every-Angle.aspx> (<http://bit.ly/TFvxmG>)
- **Tips and tricks for interpreting CHECKDB output**
 - <http://www.sqlskills.com/blogs/paul/post/CHECKDB-From-Every-Angle-Tips-and-tricks-for-interpreting-CHECKDB-output.aspx> (<http://bit.ly/TFxWOj>)



77

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

Blog Posts (2)

- **Log rebuilding and repair**
 - <http://www.sqlskills.com/blogs/paul/post/Corruption-Last-resorts-that-people-try-first.aspx> (<http://bit.ly/RRRqil>)
- **Page checksums and SQLIOSim**
 - <http://www.sqlskills.com/blogs/paul/post/How-to-tell-if-the-IO-subsystem-is-causing-corruptions.aspx> (<http://bit.ly/g4qzwp>)
- **EMERGENCY mode repair**
 - <http://www.sqlskills.com/blogs/paul/post/CHECKDB-From-Every-Angle-EMERGENCY-mode-repair-the-very-very-last-resort.aspx> (<http://bit.ly/TFy4gS>)
- **Re-attaching a suspect database**
 - <http://www.sqlskills.com/BLOGS/PAUL/post/TechEd-Demo-Creating-detaching-re-attaching-and-fixing-a-suspect-database.aspx> (<http://bit.ly/b1HSs5>)



78

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

Questions?

