SQL Server: Detecting and Correcting Database Corruption

Module 8: Simple Repair Techniques

Paul S. Randal
http://www.SQLskills.com/blogs/paul/
Paul@SQLskills.com

pluralsight
hardcore developer training

Introduction

- Usually you'll use repair because you do not have the backups necessary to restore with no data loss
- Be careful using repair or manually fixing corruptions as it may involve data loss
- Practice using repair or manually fixing corruptions before doing it for real in production

- In this module we'll cover:
  - How repair works
  - Repair options
  - Manually fixing some corruptions

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*…

Running Repair

- **Not always a last resort compared to restore**
  - Depends what backups are available
  - Depends if downtime is more important than data loss
- **All repair options require SINGLE_USER mode, i.e. they're offline**
- **Specifying a repair option makes DBCC CHECKDB run single-threaded**
- **Try to repair smallest thing**
  - DBCC CHECKDB is largest, then
  - DBCC CHECKTABLE, then
  - DBCC CHECKALLOC
- **You can put a transaction around your repair operation and roll it back if you don't like what repair did**
- **Note: most system tables can't be repaired successfully**
  - We'll cover that in the advanced course

Repair Options

- **DBCC CHECKDB will tell you which repair option is necessary**
- **REPAIR_FAST**
  - Does nothing and exists for backwards compatibility purposes
- **REPAIR_REBUILD**
  - Performs repairs that will not cause data loss
  - E.g. rebuilding damaged nonclustered indexes
    - Usually better to manually rebuild indexes, as we'll see later
- **REPAIR_ALLOW_DATA_LOSS**
  - Performs repairs that are likely to cause data loss

Beware of REPAIR_ALLOW_DATA_LOSS

- **It was very deliberately named**
- **It usually fixes structural inconsistencies by de-allocating**
  - This is the fastest and most provably correct way
- **It doesn't take into account:**
  - Foreign-key constraints
  - Inherent business logic and data relationships
  - Replication
- **Before running repair, protect yourself**
  - Take a backup and quiesce replication topologies involved
- **After running repair, check the data**
  - Run DBCC CHECKDB again to make sure all corruptions were repaired
  - Run DBCC CHECKCONSTRAINTS if necessary
  - Reinitialize any replication topologies involved

Examples of Repairs

- **What does repair do to fix:**
  - A missing nonclustered index row
    - Just insert the missing record
  - A corrupt data record
    - Maybe delete the record, maybe delete the whole page
  - An extent allocated to multiple objects
    - Performs 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

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 run repairs online (no)
- REPAIR_REBUILD will fix everything (no)
- Repair fixes up constraints (no)
- Repairs are propagated to replication subscribers (no)

Manually Fixing Nonclustered Indexes

- **It doesn’t make sense to put the database offline and run DBCC CHECKDB or DBCC CHECKTABLE with REPAIR_REBUILD to fix corrupt nonclustered indexes**
- **All it will do is essentially disable and rebuild the index, so why not do it yourself?**
- **You cannot just rebuild the index**
  - Online rebuild reads the old index to build the new index
  - Offline rebuild does that from SQL Server 2008 onward
- **Steps to use, inside a transaction:**
  - ALTER INDEX name ON tablename DISABLE
  - ALTER INDEX name ON tablename REBUILD
- **Using a transaction is necessary to prevent any index-enforced constraints from being violated while the index is disabled**

Manually Fixing Data Purity Errors

- **If a 2570 data purity error was found, SQL Server cannot repair it**
  - The error is that a column value is out-of-bounds
  - Which value should SQL Server pick to repair it?
- **You must manually repair data purity errors**
  - Work out which row is invalid using SELECT or DBCC PAGE
  - Update the column to a valid value
  - http://support.microsoft.com/kb/923247 explains the options
- **Make sure you choose a value that makes sense for your business logic and the data the column value represents**

If You Are Forced To Use Repair…

- **That implies that your backup strategy does not allow you to meet your downtime and data loss Service Level Agreements**
- **Update your backup strategy!**
  - Figure out what restores you need to be able to perform
  - Change the backup strategy to perform the backups that will allow those restores to take place
  - Implement regular backup validation
- **Also make sure that:**
  - You check any constraints that may be affected based on which tables were repaired
  - You check to see what data was lost
  - You reinitialize any affected replication topologies
  - Perform root-cause analysis

Course Summary

- **Now you know:**
  - How corruptions occur
  - How to enable automatic page corruption detection
  - How to run consistency checks
  - How to interpret DBCC CHECKDB output at a basic level
  - How to implement basic restore techniques
  - How to implement basic repair techniques
- **The most important takeaways from this course:**
  - Corruptions happen!
  - The more you practice, the easier corruption recovery will be
- **Check out *SQL Server: Advanced Corruption Recovery Techniques***
- **Email me interesting corruption stories: Paul@SQLskills.com**
- **Thanks for watching!**