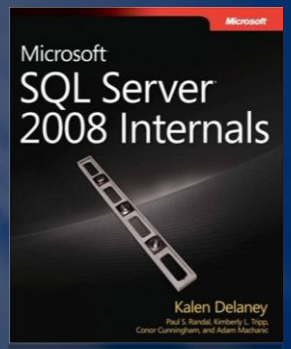
Essential Database Maintenance

Cleveland SQL Server User Group: August 2009

Paul S. Randal
Kimberly L. Tripp

SQL skills

Author/Instructor: Paul S. Randal



- Consultant/Trainer/Speaker/Author
- Managing Director, www.SQLskills.com
  - Email: Paul@SQLskills.com
  - Blog: http://www.SQLskills.com/blogs/Paul
  - Twitter: @PaulRandal
- Contributing Editor of TechNet Magazine, author of the SQL Q&A column, bi-monthly articles on DBA topics, multiple 2008 whitepapers
- 5 years at DEC responsible for the VMS file-system and check/repair
- Almost 9 years as developer/manager in the SQL Storage Engine team through August 2007, ultimately responsible for Core Storage Engine
  - Wrote DBCC INDEXDEFRAG, DBCC SHOWCONTIG and rewrote DBCC CHECKDB and repair for SQL Server 2005
- Regular presenter at worldwide TechEds and other conferences on disaster recovery, HA, maintenance, and internals
- Wrote and presented SQL Server 2008 training for Microsoft
- Course author/instructor for *Microsoft Certified Master* qualifications
- Co-Chair of the SQL Connections conference

©SQLskills.com

Author/Instructor: Kimberly L. Tripp

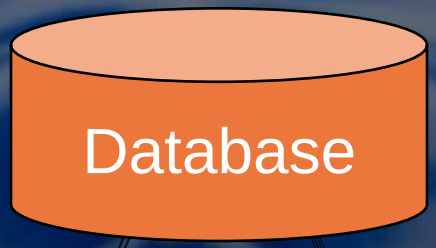
- Consultant/Trainer/Speaker/Writer
- Founder, ***SYSolutions, Inc.*** (www.SQLskills.com)
  - e-mail: Kimberly@SQLskills.com
  - blog: http://www.SQLskills.com/blogs/Kimberly
- Microsoft Regional Director and SQL Server MVP
- Writer/Editor for SQL Magazine www.sqlmag.com
- Author/Instructor for SQL Server courses: Designing for Performance, Indexing for Performance, Maintenance, Disaster Recovery/HA
- Author/Manager of SQL Server 2005 Launch Content, Co-author/Manager for SQL Server 2008 Jumpstart Content, Author/Speaker at TechEd, SQL Connections, SQLPASS, ITForum, Conference Co-chair for SQL Connections
- Author of several SQL Server Whitepapers on MSDN/TechNet: Partitioning, Snapshot Isolation, Manageability and SQLCLR for DBAs
- Author/Presenter for more than 25 online webcasts on MSDN and TechNet (two series and other individual webcasts)
- Co-author MSPress Title: SQL Server 2000 High Availability, SQL Server 2008 Internals, and the SQL Server MVP Project
- Presenter/Technical Manager for SQL Server 2000 High Availability DVD and various 2005/2008 HOL DVDs
- I still love this stuff… don't hesitate to ask questions!

The Top-Ten List…

10. Data file management

Database Structure

Up to 32,767 databases per instance

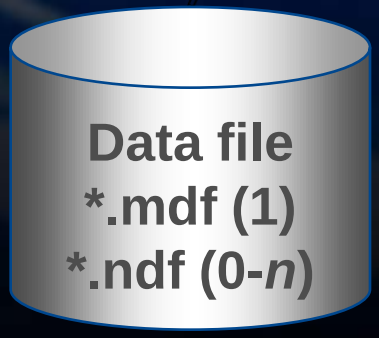


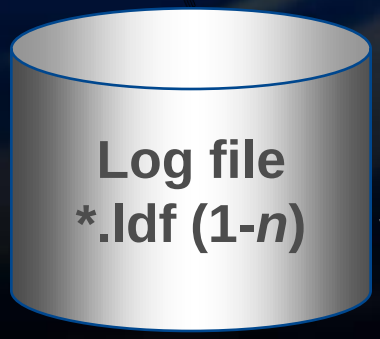
Up to 32,767 files per database
(Total is for both data and log files)

Minimum of 2 files – one for data, one for log

MDF =
Primary (**M**AIN) **D**ata **F**ile

NDF =
Non-Primary **D**ata **F**ile





LDF = **L**og **D**ata **F**ile

Should be only 1 log file

- Just because you can, doesn't mean you should…

©SQLskills.com

Capacity Planning

- Data
  - Rough estimate of current tables – be sure to calculate:
    - Base tables (8096/average row size -> rows/page * rows -> pages * 8K
    - LOB data (XML, images (varbinary(max)), etc.
    - SQL Server 2008 FILESTREAM (instead of LOB inside the database)
  - Estimate growth for 1-2 years (are you going to keep all of this data and/or periodically archive)
  - Indexes – estimate at 2-3 times your data
  - Space for performance (e.g. maintenance)
    - 1.5 times your largest table as a rough estimate
- Log
  - Recovery model
  - Disaster recovery strategy (frequency of transaction log backups)
  - Workload type – and large transactions (log cannot be cleared until they complete)

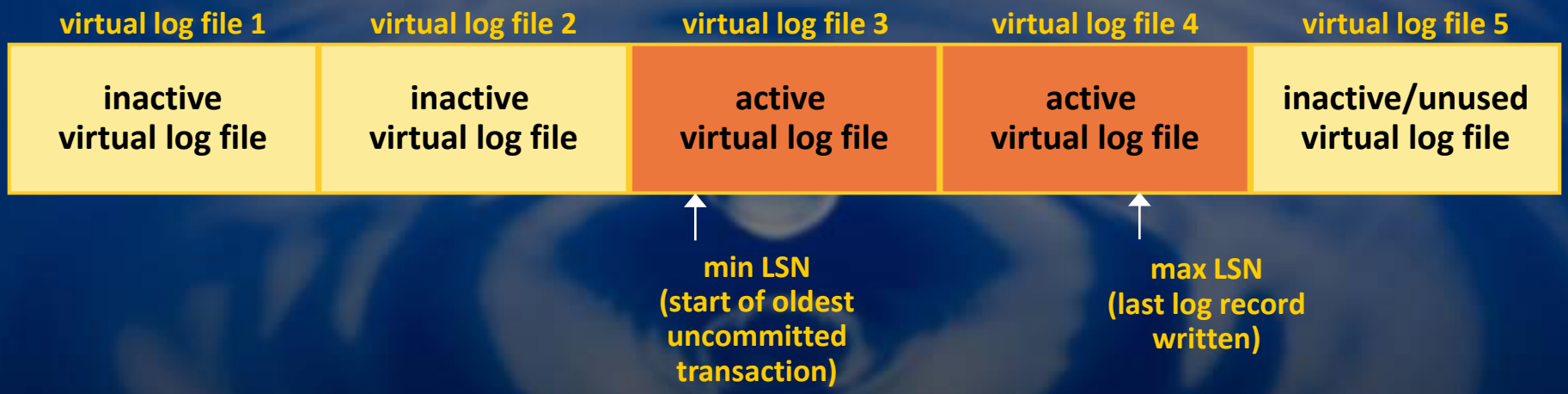
Optimizing Data Files

- Defrag the physical disks
- Effective RAID array configuration
- Pre-allocate to a reasonable initial size (minimal amount of capacity planning)
- Don't let auto-growth get out of control (monitoring)
- Turn on instant initialization
- Turn off auto-shrink (and use shrink sparingly)
- While a lot of these will help for the log as well, there are more important things to be aware of IN the data portion – tables and indexes
  - Index fragmentation

The Top-Ten List…

10. Data file management
9. Transaction log file management

How the Transaction Log Works (1)



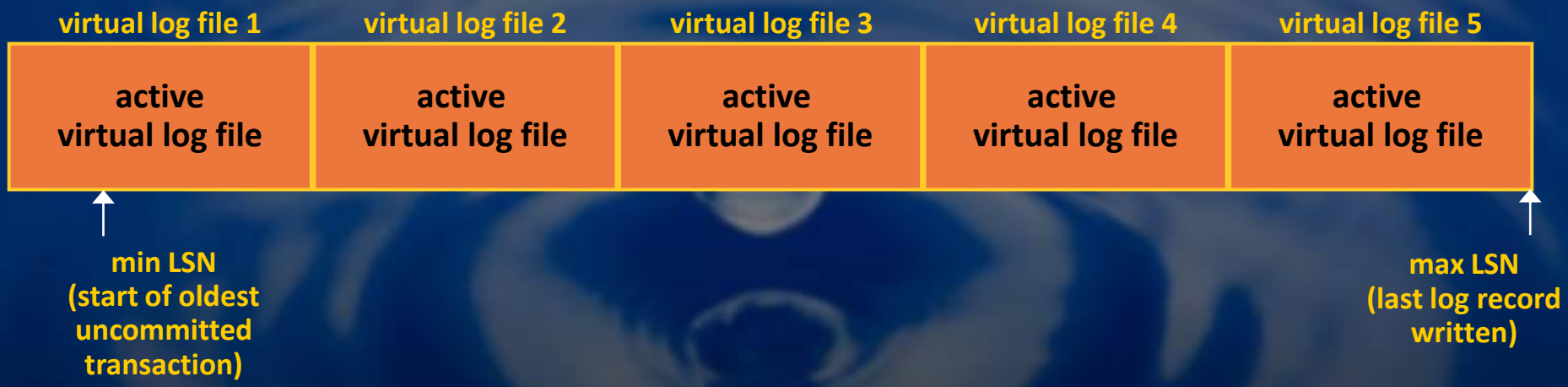
- Once a VLF no longer contains log records that are required, it can be *cleared* (sometimes called *truncated*)
- This is done by a log backup in FULL or BULK_LOGGED recovery models, and by a checkpoint in SIMPLE recovery model
- All that happens is that a VLF is marked as inactive
  - Nothing is zeroed, removed, overwritten
  - The log file size does NOT change ("truncation" is a misnomer)
- Only whole VLFs are marked as inactive

©SQLskills.com

Minimize Impact of Logging

- Log cannot clear until the transactional information is inactive
- Restart recovery will take longer if there were a lot of long running transactions pending at the time of failure
- Always try to avoid long running transactions
- Could also be:
  - Log backup, database mirroring, transactional replication, log shipping, full/differential backup, change-data-capture in 2008
- Check `log_reuse_wait_desc` in `sys.databases` for the reason the log can't be cleared

How the Transaction Log Works (2)



- What if all VLFs are active and more log records are written?
- The log will auto-grow (if configured)
  - More VLFs will be added and log records will be written into them
  - Log auto-grow means a pause while the new area is zeroed
- If the log cannot auto-grow, the database stops
  - Until the log is grown, another log file is added, or some action is taken to clear the log
- Avoid this situation by allowing the log to clear!

©SQLskills.com

If the Log File Fills Up…

- Check in `sys.databases`
- Take whatever corrective action can be done:
  - Take a log backup
  - Kill long-running transaction
  - Alter database mirroring or replication to allow the log to clear
  - Add another log file
  - Switch to the SIMPLE recovery model
    - Logged operation in 2000, not logged in 2005+
- Take preventative action to stop it happening again

Log File Considerations (1)

- Only ONE log file necessary
  - Log activity is NOT parallelized over multiple files so NO performance gain from multiple files
  - Only time another log file may be needed is if the log file runs out of space and a backup/auto-grow cannot occur
- Isolate the log file from the data files to avoid disk head contention and performance issues
- Choose an appropriate RAID level
  - RAID 5 is not recommended
  - RAID 1 is good, RAID 10 is better

Log File Considerations (2)

- Pre-allocate the log to an appropriate size
  - What is 'appropriate'?
  - Depends on your workload, recovery model, log backup schedule, HA technologies
- Log file cannot be instant initialized
  - Creation/growth takes time because of zero-initialization
- Monitor size to avoid 'runaway log file'
  - Allow auto-grow 'just in case' but do not rely on it
  - Every auto-grow adds more VLFs – balance between too few and too many
    - Too few means each will be very large and affect log clearing
    - Grow/auto-grow in no larger than 8GB chunks (i.e. 16 x 512MB VLFs)

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb

tempdb is a Special Case

- tempdb has always been used for:
  - Worktables (sort, hash tables, order by, …)
    - Query execution operations spilling out to disk
  - User created temporary options – table variables (@) and temp tables (# and ##)
  - DBCC CHECKDB processing of user databases
- tempdb in SS 2005+ is used by many new features
  - Online index operations
  - MARS (multiple active results sets)
  - Snapshot isolation
  - Triggers (INSERTED and DELETED tables)

©SQLskills.com

Contention in tempdb

- Tempdb is very susceptible to contention issues because: there's only one tempdb per instance to support all user database operations
- Some query workloads cause multiple concurrent threads to repeatedly create/drop small temp tables
- Causes huge amounts of latch contention on allocation bitmaps and system catalogs
  - SGAM page to manipulate mixed extents
  - PFS page to allocate/deallocate pages (IAM and data)
  - System catalogs to record
- Contention occurs in all versions, but worse in SQL Server 2000
- (This can sometimes (rarely) happen in user databases with VERY high-end workloads)

tempdb Allocation in 2005+

- Tempdb allocation algorithms were tweaked in 2005
- There is now a cache of temp tables
  - When a temp table is dropped, a single data page, single IAM page, and it's metadata entry remain
  - The next temp table creation pulls one out of the cache instead of having to create it from scratch
  - Alleviates some contention, but not a very big cache
- Tempdb contention can (and does) still happen in 2005 and 2008

©SQLskills.com

Alleviating tempdb Contention

- Trace flag 1118 (discussed in KB 328551) turns off the use of mixed extents in tempdb ONLY
  - Removes SGAM contention
  - Still may be needed in 2005+, but not as much as on 2000
- Creating multiple data files to spread the contention
  - For 2000, # data files should be equal to # processor cores
  - For 2005+, you can still do 1:1 but you may not need to
    - Start with # data files = ¼ to ½ # processor cores and add more if needed
    - Anecdotal evidence that going beyond 8 data files in 2005+ gives no more gain
- Note: this does not apply to log files
- Separate tempdb data and log files from other databases

©SQLskills.com

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?

©SQLskills.com

Finding the Right Balance

Index Strategies

- Determine primary usage patterns of the table
  - OLTP – fewer indexes
  - OLAP – more and wider indexes
- Create the clustering key
- Create constraints – primary key and alternate/candidate keys
- Manually add indexes to foreign key columns
- Capture a workload(s) and analyze with Database Tuning Advisor
- Add additional indexes to help improve SARGs, joins, aggregations

Are you done? **NO!**

©SQLskills.com

Clustering Key Choices

- Identity column
- Composite key of (date, identity)
  - Not date alone as not unique
- GUID
  - Populated by client-side call to .NET client guid generator
    - NOT as the CL key but OK as a non-clustered primary key
  - Populated by server-side newid() function
    - No pattern
  - Populated by NEW server-side newsequentialid() function
    - Creates a better (but not ideal) pattern

Key point: Unique, narrow, static and ever-increasing!

©SQLskills.com

Indexing Best Practices

- Don't go crazy adding indexes
- Just because you can index every column (and/or INCLUDE every column) – doesn't mean you should!
- Stop indexing after you've setup the base table structures and start assessing where to go from here…
  - Over-indexing can be worse than under indexing
  - Indexes that are not maintained can be more trouble in the long run (setup maintenance)
  - Remember, a narrow index has very few uses…
  - A wider index has many more uses – you might be able to make some of your existing indexes more useful by just making them wider (ok, you can't directly add columns but you can create new indexes and then drop the ones you're not using)

Indexing for Performance

Improving Performance in an Existing Workload

1. Get rid of any unused indexes before you do anything…
   1. Prune redundant indexes
   2. Remove unused indexes (sys.dm_db_index_usage_stats)
2. Verify the health of the existing (and useful) indexes…
   1. Determine the level of fragmentation (sys.dm_db_index_physical_stats)
   2. Determine splits/latch waits (sys.dm_db_index_operational_stats)
3. Add indexes to improve poorly performing queries?
   1. Ask DTA what it thinks?
   2. Ask SQL Server what it thinks?
      1. SQL Server 2005: DMVStats, Performance Dashboard Reports (SP2+), RML Utilities from PSS
      2. SQL Server 2008: [Performance] Data Collector
      3. Both SQL 2005/2008: DMV queries (missing index DMVs)
         Bart Duncan's blog: (search "Bart Duncan missing")

©SQLskills.com

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics

Statistics: Why are they important?

- For indexes to be used, the optimizer needs to be able to estimate their cost (to choose the lowest expected cost)
- Cost-based optimization is done by assessing the usefulness of an index by looking at:
  - The query submitted
  - The conditions used (search arguments, join conditions, group by, order by, etc…)
  - The selectivity of the data (based on the arguments)
- Selectivity is determined by statistics
  - Statistics on indexes are of primary importance
  - Statistics on the secondary columns of indexes can provide some interesting uses…

©SQLskills.com

Statistics: When are they Created?

- Statistics are automatically created
  - For all indexes
  - For other columns, when the database option “auto create statistics” is ON AND when the optimizer thinks that statistics would be a good idea
- Statistics can be created manually
  - sp_createstats
  - Using CREATE STATISTICS
- Tips
  - Leave the database option: auto create statistics on
  - Add more statistics
    - SQL Server 2000 supports 249 total indexes/stats per table
    - SQL Server 2005 supports 2,000 statistics **per table**
    - SQL Server 2008 supports 10,000 statistics **per table**

©SQLskills.com

sp_createstats

```
sp_createstats @indexonly = 'indexonly',
   @fullscan = 'fullscan',
   @norecompute = 'norecompute'
```

- @indexonly: only create statistics for the secondary columns of indexes. This can help to make non-clustered indexes more useful.
- @fullscan: requires more time but will create more accurate statistics. If off-hours this is a good idea
- @norecompute: the statistics will NOT get automatically updated as distribution of data changes. Generally, not recommended.
- **Recommendation:**

```
sp_createstats 'indexonly', 'fullscan'
```

©SQLskills.com

Statistics: What if the data changes?

- Automatically updated
  - If auto update statistics is ON (for both the DB and the index wasn't created with "norecompute")
  - If a percentage of the data changes
    - Complete details in TechNet whitepaper
- Manually update statistics
  - For highly volatile tables where distribution isn't changing significantly and you see a lot of "statistics" events
  - Turn off "auto update" OR turn off auto update by adding STATISTICS_NORECOMPUTE on the index definition (better control)
  - Executing UPDATE STATISTICS

Auto Update Statistics

- SQL Server 7.0
  - Invalidated when sysindexes.rowmodctr reached
  - Updated when invalidated
- SQL Server 2000
  - Invalidated when sysindexes.rowmodctr reached
  - ✓ Updated when needed
- SQL Server 2005
  - ✓ Invalidated when sysrowsetcolumns.rcmodified reached
  - Updated when needed
- SQL Server 2008
  - Invalidated when sysrscols.rcmodified reached
  - Filtered statistics

Review the Statistics Used by the Query Optimizer in Microsoft SQL Server 2005 whitepaper.

©SQLskills.com

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics
5. Index fragmentation

©SQLskills.com

Fragmentation Defined

- Two broad types:
  - Internal fragmentation
    - Wasted space on data/index pages
    - Sometimes called ‘physical’ fragmentation
  - Logical fragmentation
    - Out-of-order pages within an index leaf level affecting range-scan performance
- Both can have very detrimental effects on performance

©SQLskills.com

Page Density (Internal Fragmentation)

- Page fullness is below the required fillfactor
- More pages required to hold the same rows
- How does this happen?
  - Poor schema design leading to very large rows and wasted space on each page
  - Random deletes leaving space on pages
  - Page splits
- Effect is:
  - Increased IOs for range scans
  - Greater memory usage if most of the index is memory resident
  - Performance must be decreased because of page splits!!

©SQLskills.com

What Causes Fragmentation?

- Schemas/workloads that cause page splits on densely packed pages
  - GUID as high-order key (or any other random key)
    - Can even affect nonclustered indexes
  - Updates to variable-length columns
  - Badly configured `FILLFACTOR` (more in a few slides)
- Wide schemas that only fit a few records per page and waste space
  - E.g. a fixed-size 5000 byte row = 3000 bytes lost per page!
- Real-world example:
  - Social networking site that has a homepage comments table with the member ID as the high-order key
  - Needs fragmentation removed twice a day!

©SQLskills.com

Detecting Fragmentation

- Keys to success are:
  - Knowing which indexes to look at
    - Which are used for range scans?
    - Which have very low page density?
  - Knowing which options to use of the various methods
  - Knowing how to interpret the results
- Use sys.dm_db_index_physical_stats DMV in 2005
- Use DBCC SHOWCONTIG in 2000
  - Still there in 2005 but deprecated
- Don’t just rebuild all indexes every day!

How to Correct Fragmentation?

- 3 realistic choices
  - CREATE INDEX WITH DROP_EXISTING
  - Rebuild the index – ALTER INDEX … REBUILD
  - Defrag the index – ALTER INDEX … REORGANIZE
- Most people rebuild or reorganize
  - Trade-offs between the two methods
    - Log space required
    - Disk space required
    - Online vs. offline
    - Time to completion
    - Interruptible without loss of work

Index Rebuild Misconceptions

- Rebuilding a clustered index rebuilds all nonclustered indexes
  - No, only in 2000 and before for non-unique clustered indexes
  - There was a bug in 2000 SP1 that *did* rebuild always
- Index rebuild pre-allocates the necessary space
  - No, it allocates space as it goes so may run out
- Rebuilding in BULK_LOGGED reduces the size of the log backups
  - No, only the size of the transaction log. Remember how log backups work after a minimally-logged operation?
- Index rebuilds in a multi-file filegroup only use a single file
  - No, the allocations will follow regular round-robin proportional fill
- Online index build doesn't take any locks at all
  - No, online is a slight misnomer

©SQLskills.com

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics
5. Index fragmentation
4. Detecting corruption

©SQLskills.com

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..suspect_pages`
  - Input into single-page restore operations

©SQLskills.com

Page Checksums (1)

- Per-page checksum
  - Written as last thing SQL Server does on a physical write
  - Checked as first thing SQL Server does on a physical read
- Provides the "smoking gun" that the error is not due to SQL Server as it detects all changes to a page
  - Superset of torn-page detection
- On by default for new databases in SQL Server 2005+
- Checksum failures result in an 824 error
- CPU overhead of approx 1-2% on heavy load
- Error-detecting, not error correcting

Page Checksums (2)

- Checked when:
  - Page is read normally
  - Page is read during `DBCC CHECKDB`
  - Page is read during `BACKUP WITH CHECKSUM`
  - Page is contained within a checksum'd backup
- There are two problems though:
  - Not available for tempdb until SQL Server 2008
  - Switching it on doesn't do anything until pages are read, changed, written…
- Blog post:
  - http://www.sqlskills.com/blogs/paul/post/How-to-tell-if-the-IO-subsystem-is-causing-corruptions.aspx

DBCC CHECKDB

- The only way to read all allocated pages in the database
  - Use to force page checksums to be checked
- Many algorithms to minimize runtime and run ONLINE since SQL Server 2000
  - Compared with 6.5 or 7.0
- Variety of ways to perform consistency checks depending on resource constraints
- New features in 2005
  - Progress reporting, data purity, last known good
- 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

Consistency Checking

- How often should CHECKDB be run?
- How long will CHECKDB take to run?
- How best to consistency check a VLDB?
- What does all the CHECKDB output mean?
  - Answer to all of these is "It depends!"
- Is it better to restore or just run repair?
  - Restore usually guarantees zero data loss, but whether its faster depends on backup strategy

- Bottom line: regular consistency checks must be run to proactively find corruption
- Blog post series: http://www.sqlskills.com/blogs/paul/category/CHECKDB-From-Every-Angle.aspx

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics
5. Index fragmentation
4. Detecting corruption
3. Problem notification

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 the consistency checks 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

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics
5. Index fragmentation
4. Detecting corruption
3. Problem notification
2. Taking backups

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 required starting point
  - Series of log backups as well to reduce data loss
- Remember:
  - Backups have to exist to be useful
  - Backups have to be valid to avoid data loss

©SQLskills.com

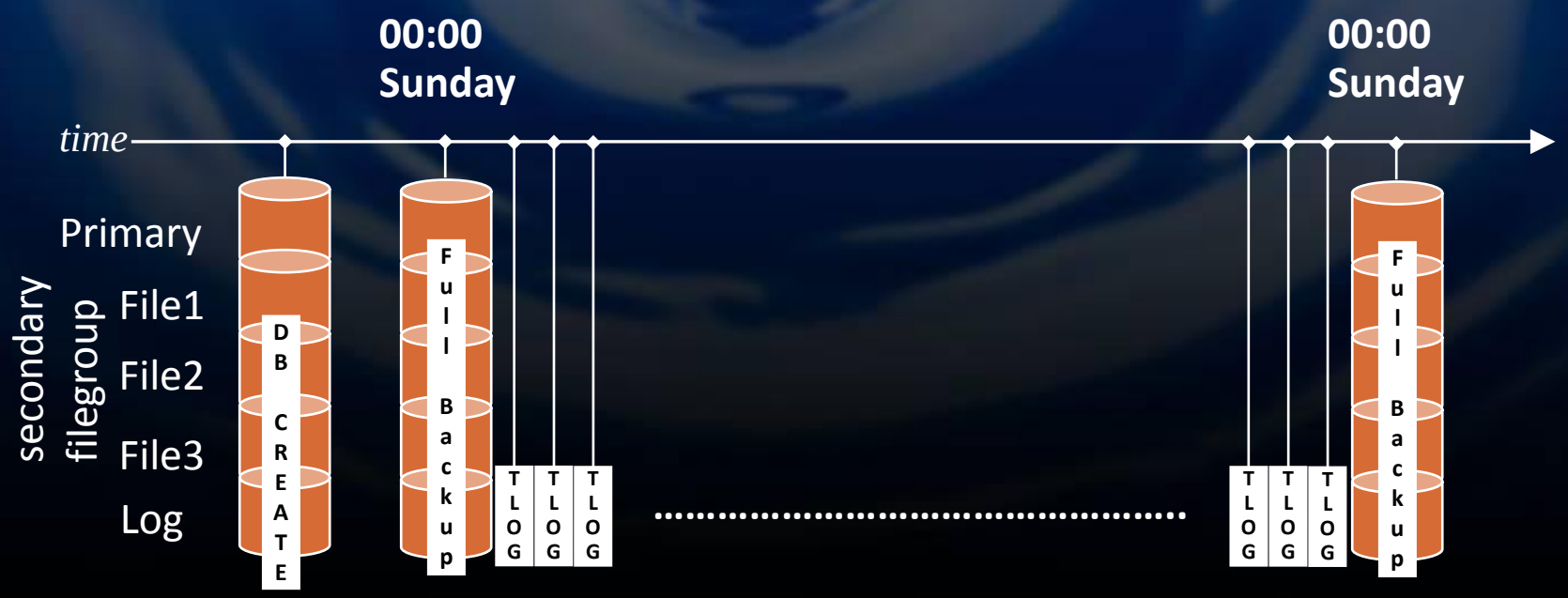
Full Database Backup Only Strategy

- Only as good as your last full backup
- Significant amount of work loss exposure
- No possibility of point-in-time recovery
  - What happens if a crash occurs at 23:59 on Saturday?

©SQLskills.com

Full + Log Backup Strategy

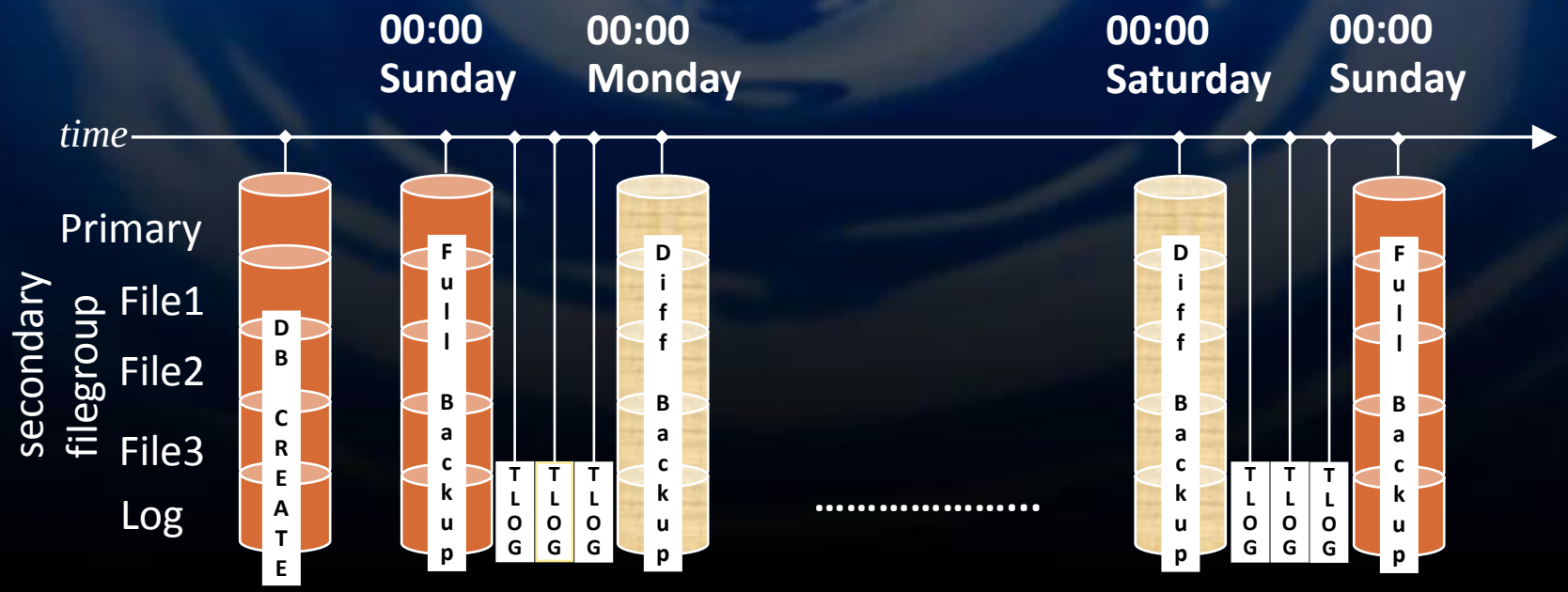
- Can get up-to-the-minute and point-in-time recovery
- Restoring a large number of logs is hard to manage and can take a significant amount of time
- May not have much work loss exposure at all
  - What happens if crash occurs at 23:59 on Saturday?



©SQLskills.com

Full + Diff + Log Backup Strategy

- Ultimate in availability, but less common
- Periodic differential backups reduce the recovery time by removing the need to apply all log backups since last full
- Does not *remove* the need for log backups
  - What happens if a crash occurs at 23:59 on Saturday?



©SQLskills.com

The Top-Ten List…

10. Data file management
9. Transaction log file management
8. Tempdb
7. Are your indexing strategies working?
6. Statistics
5. Index fragmentation
4. Detecting corruption
3. Problem notification
2. Taking backups
1. Testing

©SQLskills.com

It's All About Testing

- Test your:
  - Disaster recovery plan
  - Backups are valid
  - Indexing strategy
  - Alerts

- Test, test, test
- And then test again in a few months.
- And again.
- Did we mention testing? ☺

References

- TechNet Magazine articles:
  - Top Tips for Effective Database Maintenance
    - http://technet.microsoft.com/en-us/magazine/cc671165.aspx
  - Understanding Logging and Recovery
    - http://technet.microsoft.com/en-us/magazine/2009.02.logging.aspx
  - Understanding SQL Server Backups
    - http://technet.microsoft.com/en-us/magazine/dd822915.aspx
- Blog posts
  - Very hard to single out a few, but both of us tag our posts with categories. Send email if there's something you want to find.

Thank You!

Questions?

©SQLskills.com