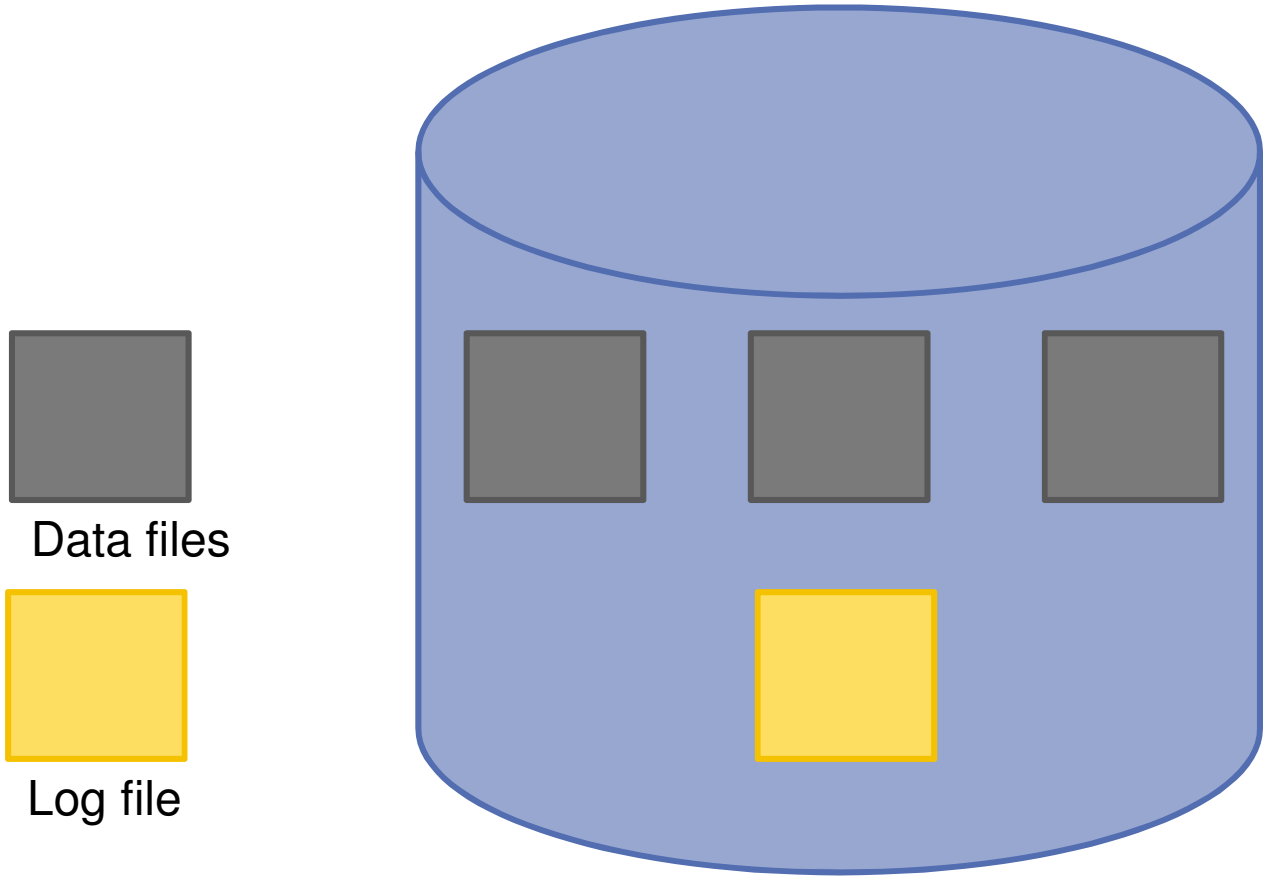
# How SQL Server Stuff Works: Log Files

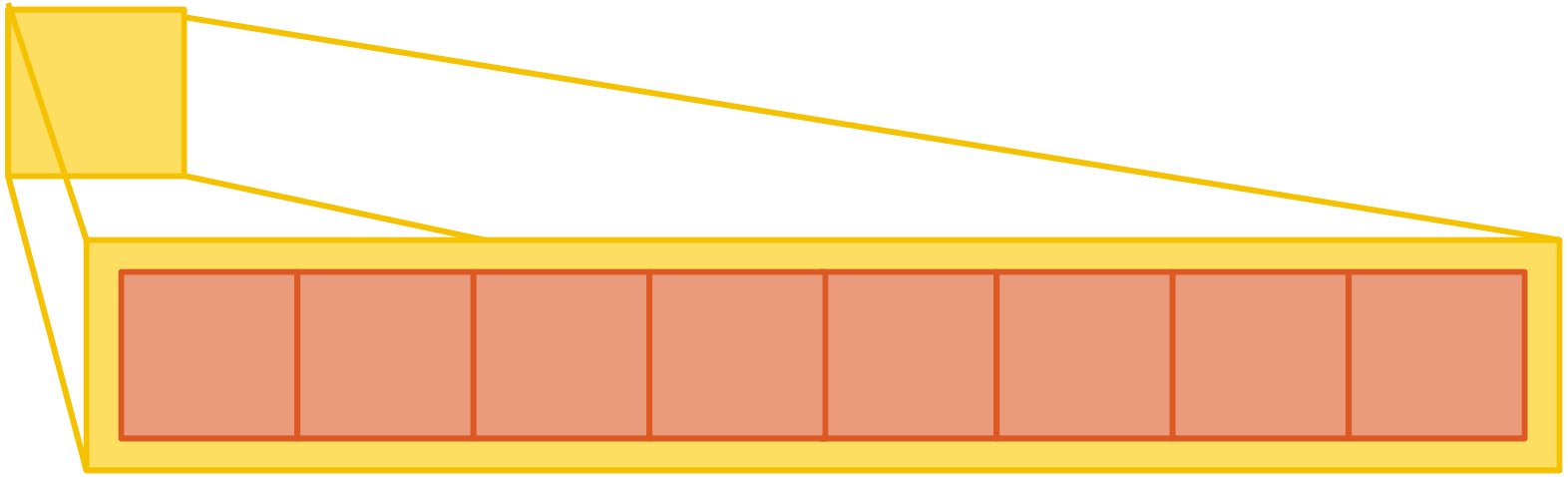
Jes Schultz Borland

Consultant, Brent Ozar PLF

# Database architecture



# Log file architecture



Virtual Log Files

- No fixed size
- No fixed number per physical file

# My favorite question: why?

**Since there is only one log file, the space needs to be reused**

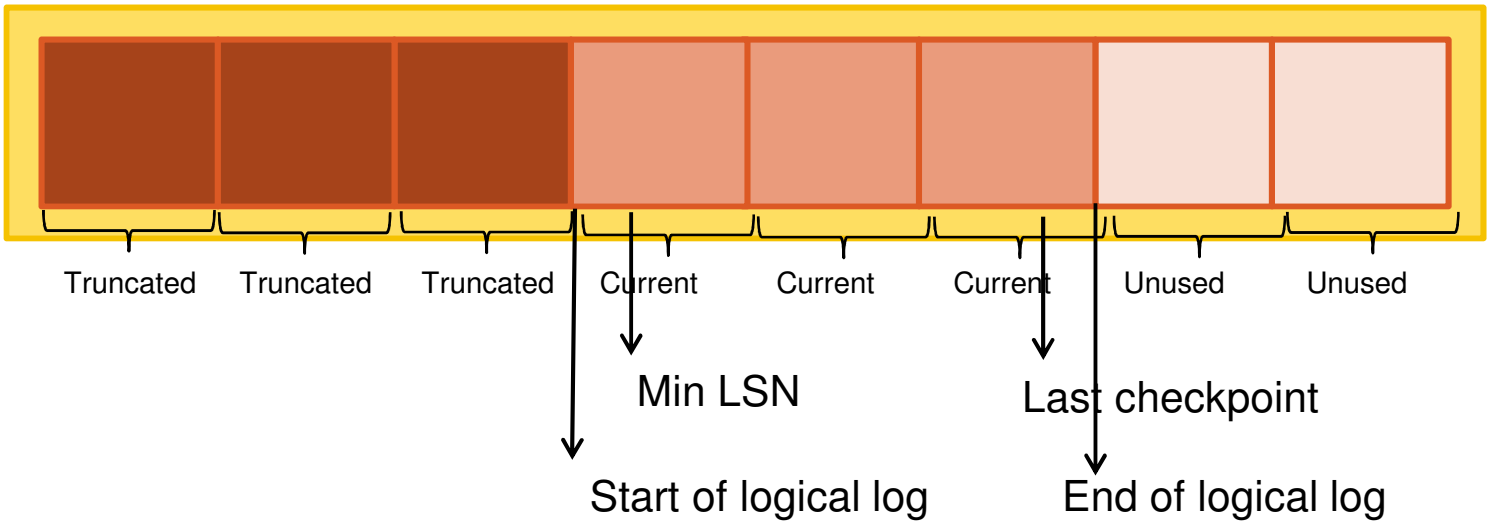
**VLFs are used to manage space reuse**

# How SQL Server writes to the log

**SQL Server begins writing at the "front" of the log file**

**As log records are truncated* (cleared), the VLFs are cleared**

**A minimum log sequence number (LSN) is maintained as the oldest log record required for a successful rollback**



# A few notes about truncation

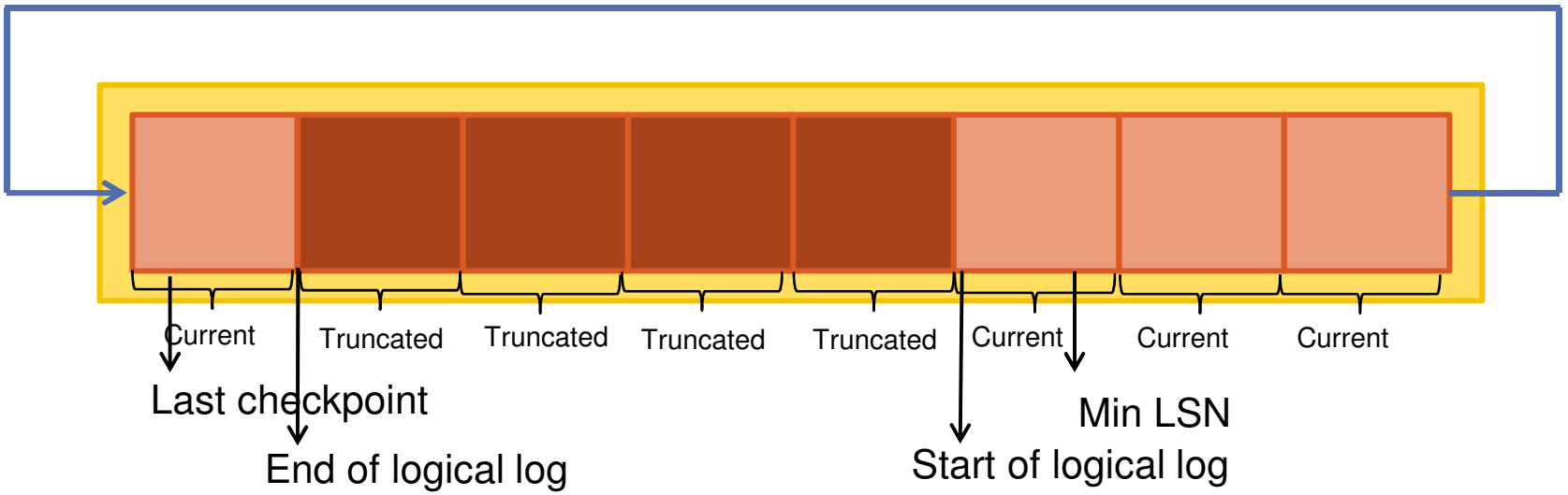
**SQL Server does not delete anything in the file**

**The space is simply marked for re-use**

**If you do not set up log backups, or have open transactions, SQL Server cannot truncate the log – and it never stops growing!**

# How SQL Server writes to the log

**When the end of the transaction log file is reached, it starts back over at the beginning of the physical file.**




**However, this can only happen if the end of the logical log never catches up to the start of the logical log.**

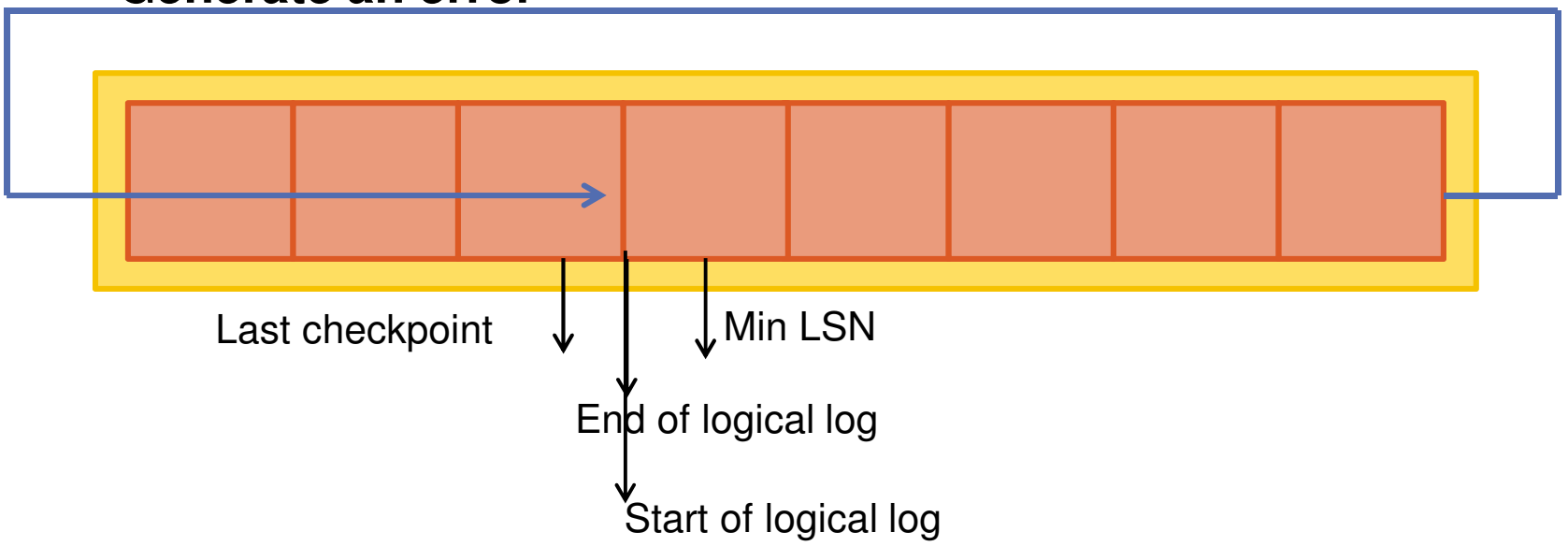
# How SQL Server writes to the log

**At this point, SQL Server will either:**

- **IF autogrow is enabled AND space is available on disk, grow the log**

**OR**

- **Generate an error**



# What we really want…

**Is the log file to grow.**

**When the physical log file grows, more VLFs are added also.**

# How many VLFs are added?

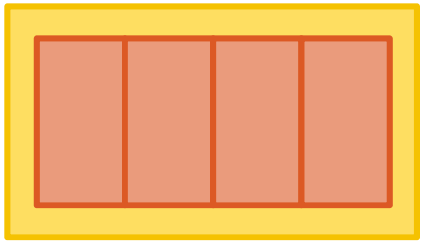
**When the log grows:**

- **Up to 64MB – 4**
- **More than 64 MB, less than 1 GB – 8**
- **More than 1 GB – 16**

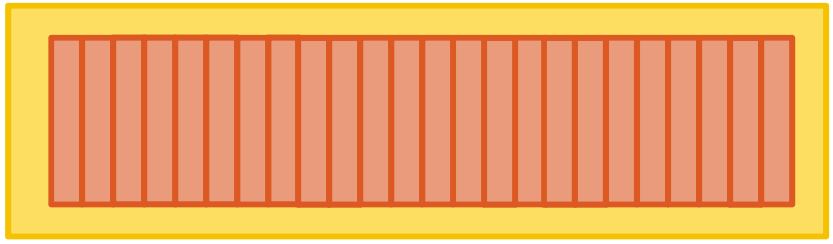
**SQL Server default file growth is 10 MB**

# Small Log with Small Autogrow

**The log starts at 50 MB. (One physical file, 4 VLFs.)**

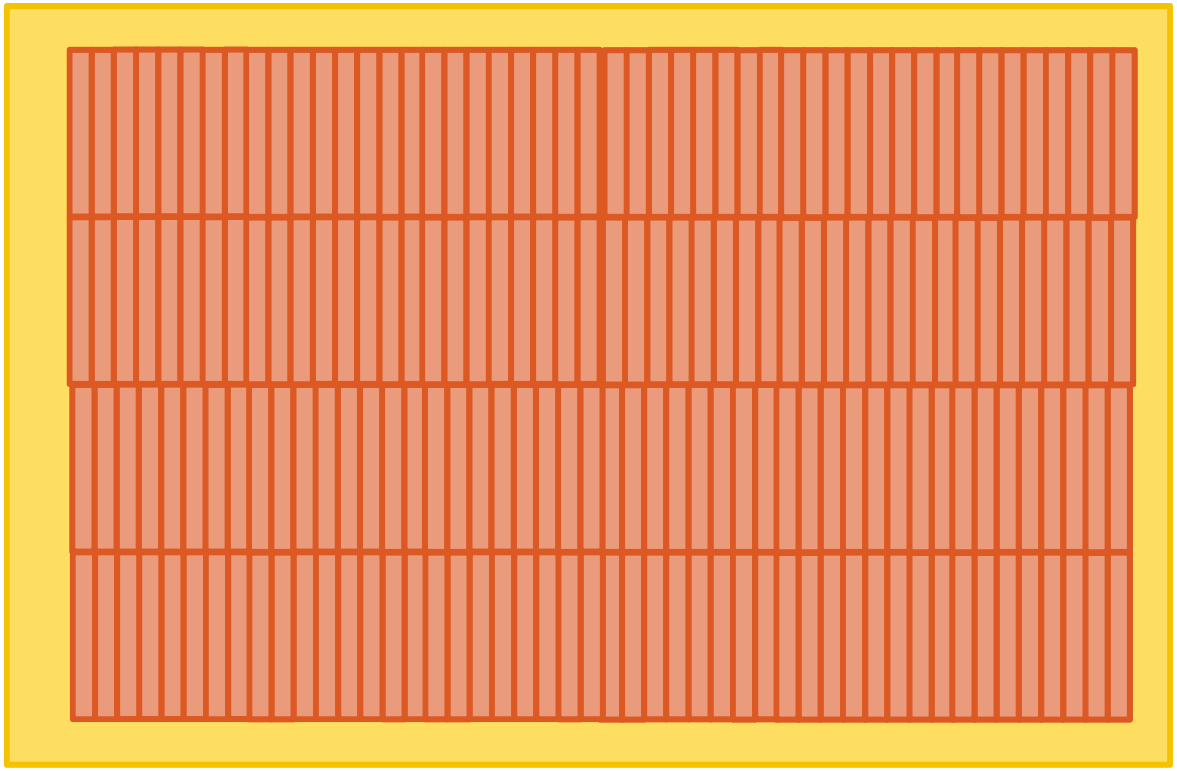


**It needs to increase to 100 MB. It grows 50 MB, 10 MB at a time, so 20 more VLFs are created, for a total of 24 VLFs.**



**What will that look like by the time the log is at 500 MB?**

# 500 MB log with 10 MB default growth



*Number of log files is approximate.

Imagine what a 50 GB log file would look like!

# Problems

**Inserts, updates, and deletes take longer**

**Starting up a database will take longer**

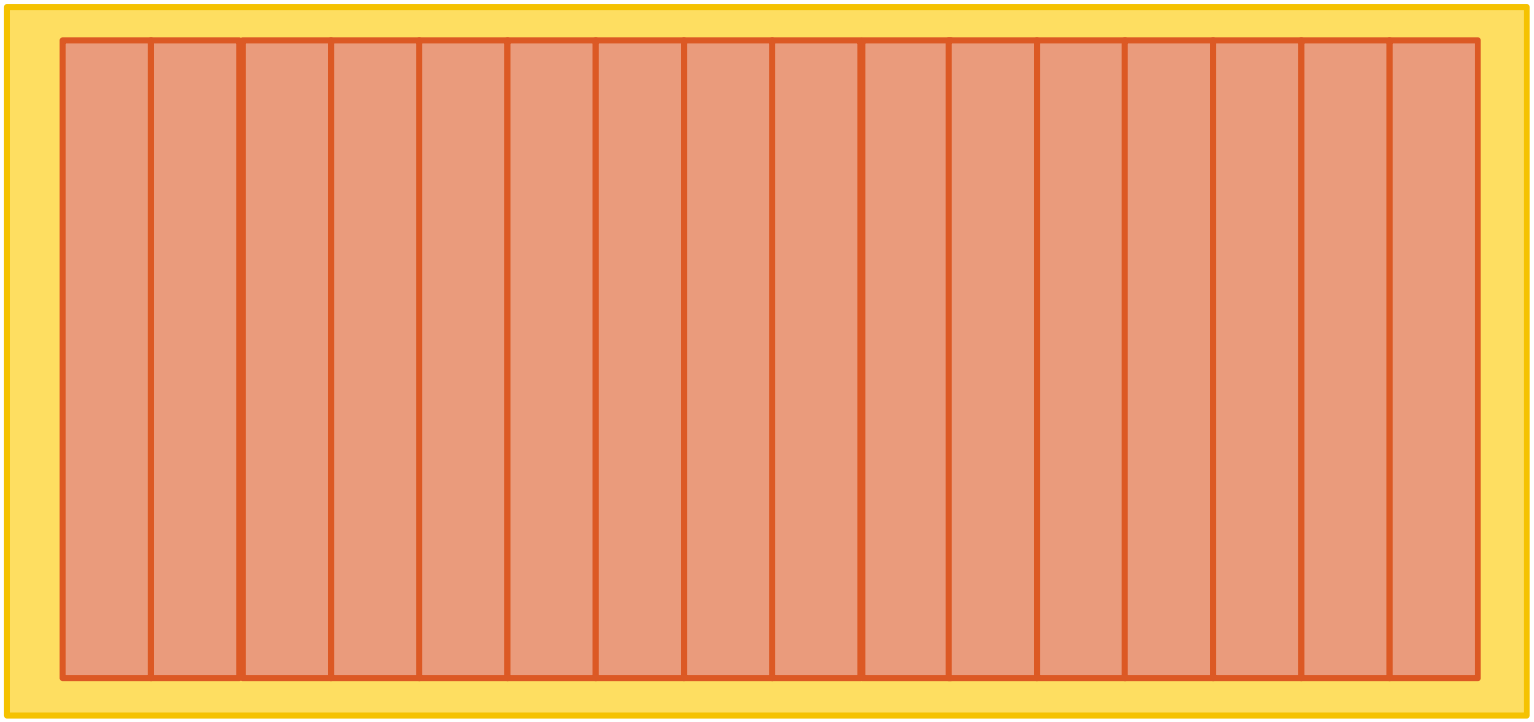
**Restoring a database will take longer**

**Linchi Shea: Performance impact: a large number of virtual log files – Part I - http://sqlblog.com/blogs/linchi_shea/archive/2009/02/09/performance-impact-a-large-number-of-virtual-log-files-part-i.aspx**

# Large log with large autogrow

**A log starts at 64 GB. (One physical file, 16 VLFs.)**

**Each VLF is 4 GB!**



# Problems

**Clearing a 4 GB VLF will take time**

# On one hand…

**Lots of growth in small increments leads to many VLFs**

**Why is this bad?**

- **Fragmentation**
- **Slow startup/recovery time**

# on the other hand…

**Small amounts of growth in large increments leads to large VLFs**

**Why is this bad?**

- **Clearing (truncating) a VLF can take a long time, provided it is not in use and can be cleared**

# How can I tell how many VLFs I have?

**DBCC LOGINFO (undocumented) returns one row per VLF**

**Dave Levy blogged a script: http://brentozar.com/go/vlf**

# Recommendations

**There is no specific formula for how many VLFs you should have in your log file.**

**Too many leads to fragmentation.**

**Too few leads to large VLFs and slow performance.**

**The goal is to have your log be as large as necessary, with a reasonable autogrowth increment (example: 128 MB).**

# Fixing Log Files with Lots of VLFs

- **Figure out appropriate log size**
- **Shrink log to smallest size possible**
  - Shrink in increments
- **Grow to appropriate size in increments**
- **Check autogrow settings**

**Consider growth strategy carefully so there are neither too many VLFs, nor too large of VLFs**