

# Transaction Log Internals and Troubleshooting

Andrey Zavadskiy  
Russia



# Sponsors

---



PLURALSIGHT



# About me

- Solutions architect, SQL & .NET developer
- 20 years in IT industry
- Worked with SQL Server since 7.0 back in 2001
- Developed in C#, ASP.NET, MVC, JavaScript, SharePoint
- MCDBA, MCSE, MCSD
- MCT since 2008
- PASS speaker



<http://andreyzavadskiy.com>



<https://www.facebook.com/andrey.k.zavadskiy>



@AndreyZavadskiy



<https://www.linkedin.com/in/zavadskiy>

# Contents

---

- Logical and physical architecture
- Transactions and transaction log
- Log file growing and truncation
- VLF fragmentation
- Troubleshooting
- Delayed durability (SQL Server 2014)

# Transaction Log. What for?

---

- Supports ACID properties
- Recovery in case of database crash or SQL Server startup
- Rolling a restored database, file, filegroup, or page forward to the point of failure
- Supporting transactional replication
- Supporting high availability and disaster recovery solutions

# Logical Architecture

- Just a list of log records
- Identified by Log Sequence Number (LSN)

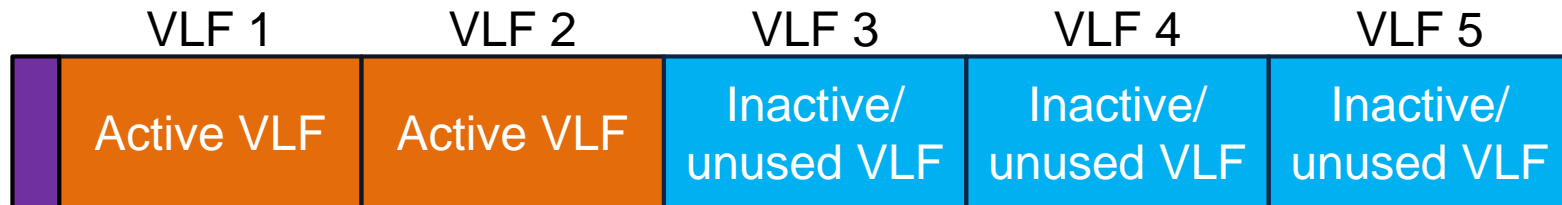
00000028 : 00000120 : 0002

↖ ↗ ↖

VLf number    Log block number    Log record number

- Log record contains info about transaction, before and after images, allocation, etc.
- Can be viewed by fn\_dblog()

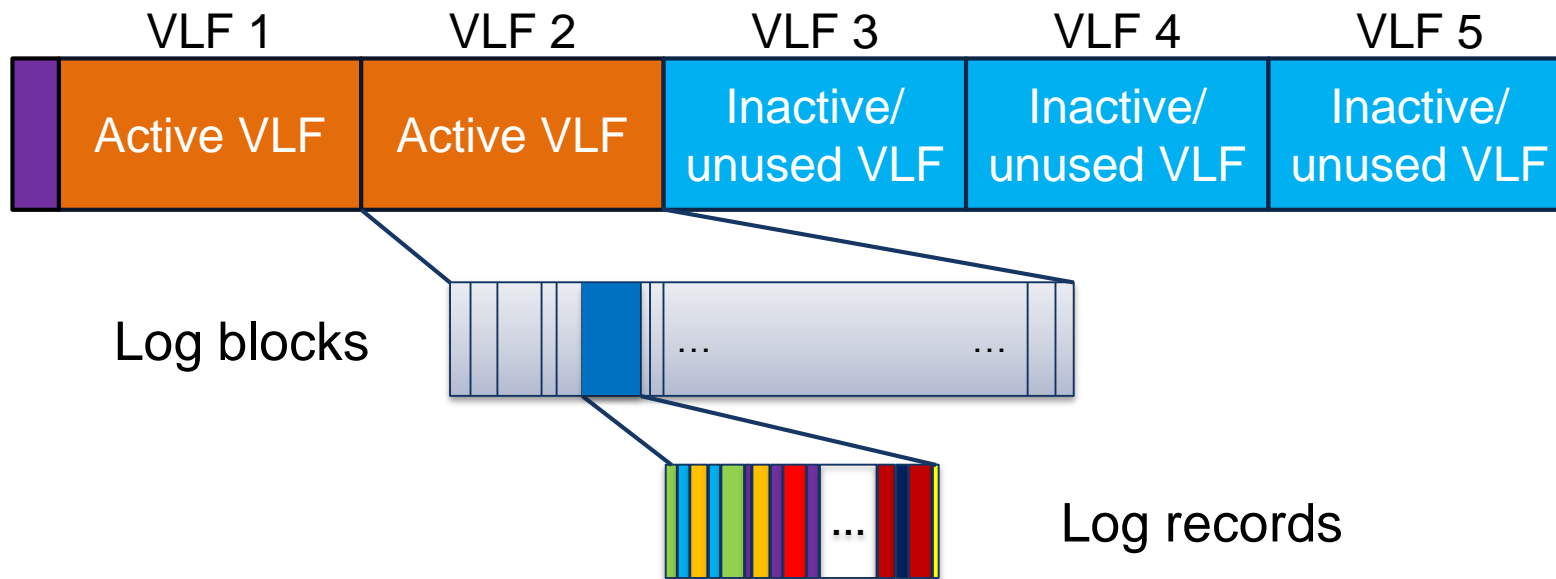
# Physical Architecture (1)



Header

- 8KB file header with metadata
- Consists of Virtual Log Files (VLF)
- VLF has logical sequence number (FSeqNo)
- Minimum 2 VLFs, minimal VLF size = 248KB
- Filled by zero on creation

# Physical Architecture (2)



- Each VLF is splitted into log blocks
- Log block size = from 512B to 60 KB
- Contains log records from multiple transactions



# Transactions and Transaction Log

---

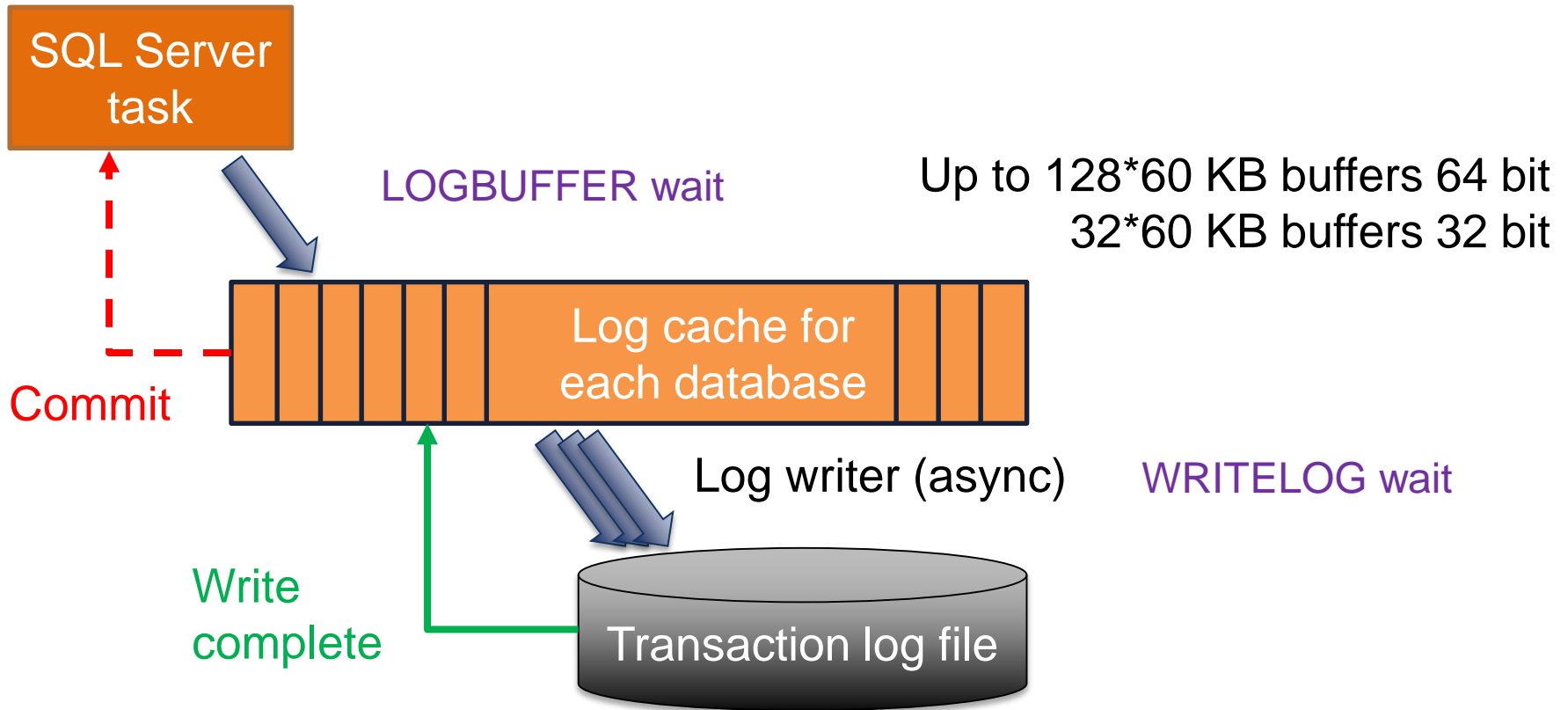
- Records all modifications made by each transaction
- Single transaction produces some log records
- Written to transaction log file
- Implements Write-Ahead Log (WAL) by default
  - Can be changed by Delayed Durability

# Transaction Commit

---

- All log records up to the LSN of LOP\_COMMIT\_XACT must be written to disk
- Waits for acknowledgement from the synchronous mirror or Availability Group server (if applicable)
- Release all locks placed by the transaction
- Acknowledge the commit to user

# Transaction Log Flush



# Transaction Log Writes

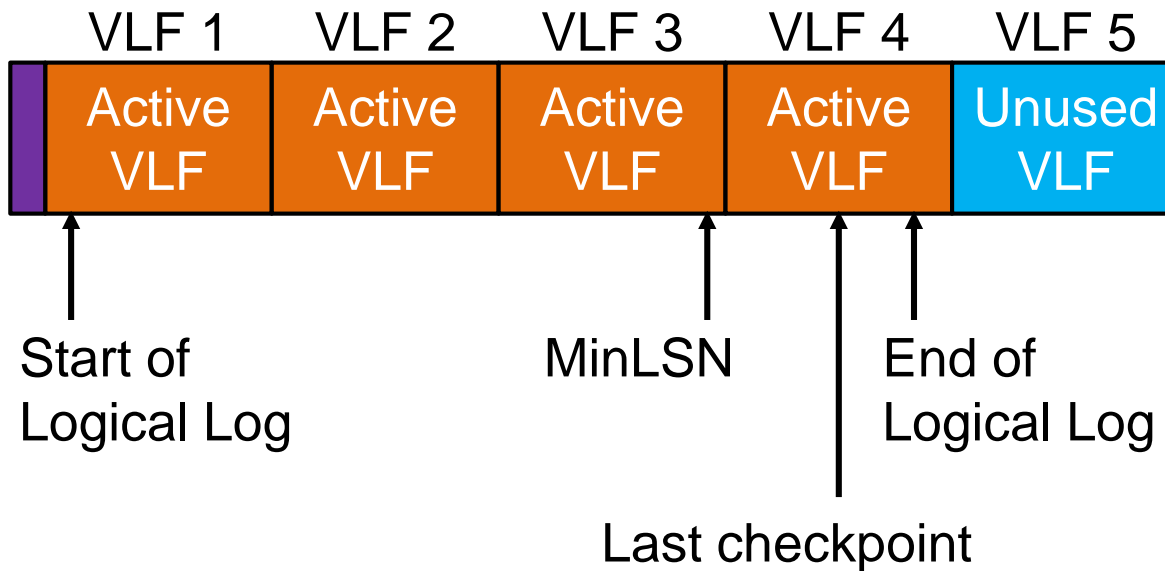
- Always writes sequentially
  - Multiple log files doesn't give any performance benefit
- Limits on outstanding I/O:

SQL Server version	Quantity		Amount, KB
	64 bit	32 bit	
Prior to 2005SP1	8	8	480
2005SP1 – 2008R2	32	8	480 (2005) 3840 (2008)
2012 and later	112	16	3840

# Log File Operations

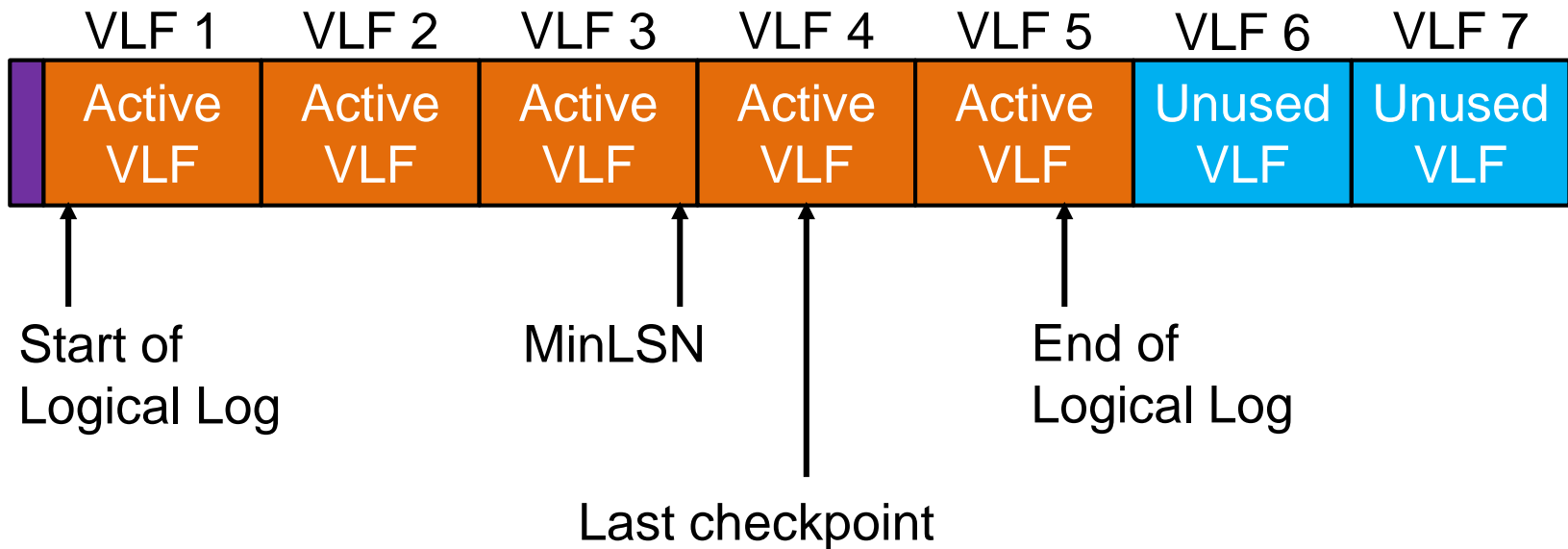
Operation	File size	Number of VLF	Comment
Growing	Increasing (expands a log file according to growth parameters)	Increasing (calculated by VLF formula)	
Truncation	Not changed	Not changed	Only marks inactive VLFs as truncated
Shrinking	Can be reduced	Can be reduced	Depends on active VLFs

# Log File Growing (1)



- SQL Server allocates so many VLFs as needed to rollback the longest active transaction
- New VLFs are filled by zero

# Log File Growing (1)



- SQL Server allocates so many VLFs as needed to rollback the longest active transaction
- New VLFs are filled by zero

# VLF Size Algorithm

- Used on log creation for all versions
- Used on log growth for SQL Server 2012 and earlier
- Depends on chunk size to be added

Chunk size	Number of VLF added
Size <= 1 MB	Complicated, playing with first 248 KB VLFs
1Mb < Size <= 64 Mb	4 VLFs
64 Mb < Size <= 1Gb	8 VLFs
Size > 1G	16 VLFs



# New VLF Size Algorithm

---

- Used only on log growth since SQL Server 2014
- Depends on chunk size to be added:
  - If the chunk size less than  $\frac{1}{8}$  of the current log size, create 1 VLF equal to the growth size
  - Otherwise, create VLFs according to the old algorithm

# Log File Growing (2)

---

- Log file has initial and maximum sizes
- Maximum size can be fixed or unlimited\*
- Log file can be expanded manually or automatically
- If log autogrowth occurs:
  - New VLFs will be added and zero-initialized
  - It leads to a wait in transaction processing

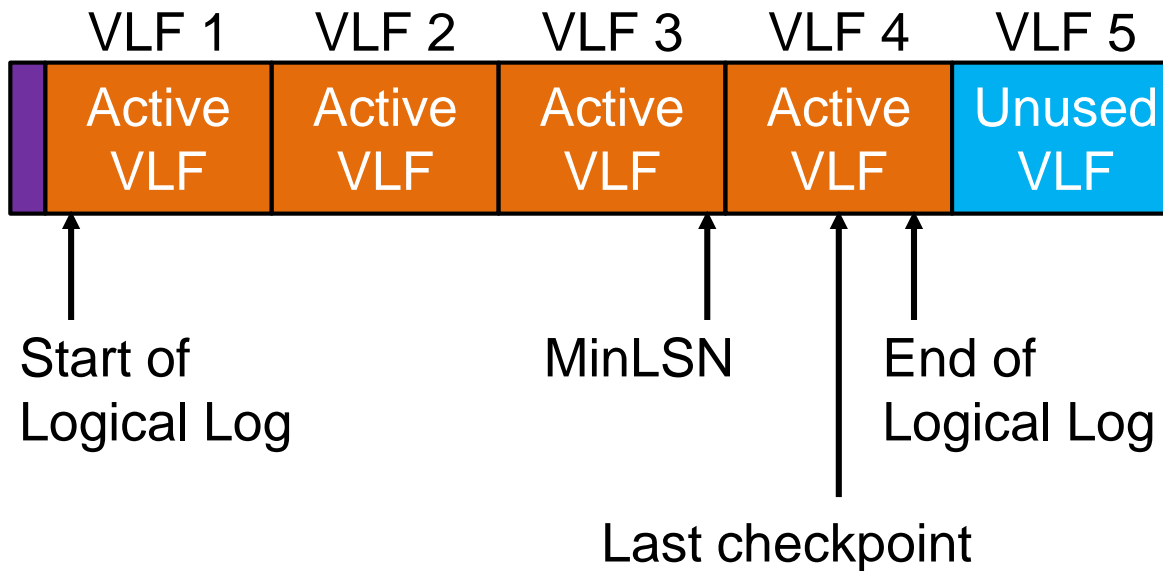
# Inactive Log Record

---

Log record becomes inactive when:

- The transaction that this log record is part of has committed
- The database pages changed by this transaction/log record have been written to disk by checkpoint
- The log record is not needed for a backup (full, differential, or log)
- The log record is not needed for any feature that reads the log (Database mirroring, AlwaysOn Group, Transactional replication, Change Data Capture)

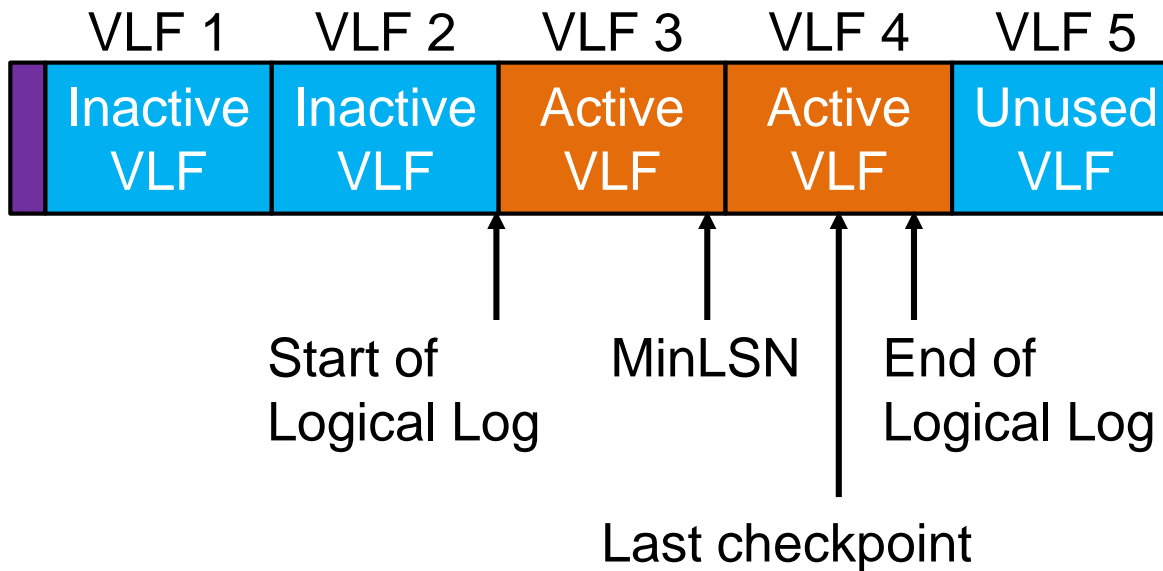
# Log Truncation (1)



VLF is truncated when:

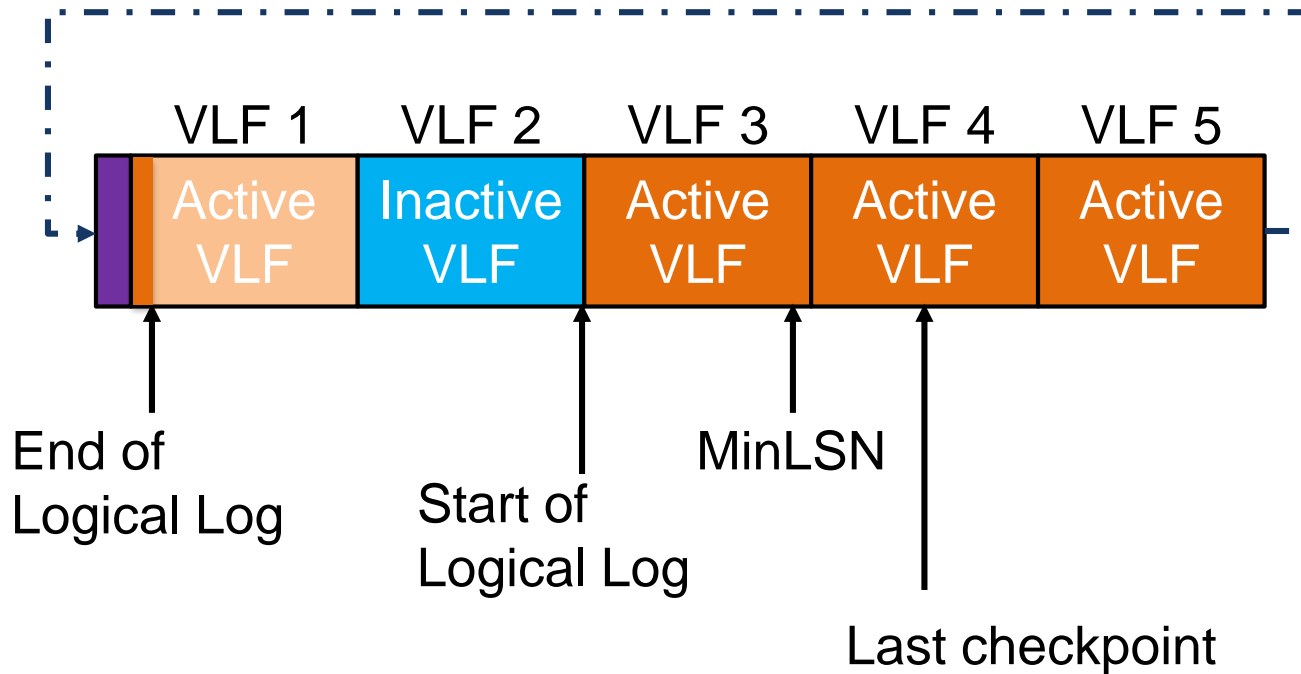
- It has NO active log records
- After checkpoint in simple/pseudo-full recovery model
- After log backup in full or bulk-logged recovery model

# Log Truncation (2)



- VLF is marked as truncated
- VLF is NOT filled with zero

# Circular Nature of the Log



- Inactive VLF can be overwritten
- Parity bits are flipped after roll-over

# Shrinking

---

- Automatic
  - Database AUTO\_SHRINK option
- Manual
  - DBCC SHRINKFILE
  - Shrink unused VLFs/space from the end of the log file
  - Could shrink maximum to the first 2 VLFs

# Transaction Log Issues

---

- Excessive log growing and error 9002
- Log shrinking
- VLF fragmentation



# Excessive Log Growing

There are many reasons

- Look at *log\_reuse\_wait\_desc* in *sys.databases*
- See section “Factors That Can Delay Log Truncation” in <https://msdn.microsoft.com/en-us/ms190925.aspx>

How to correct:

Reason	Action
LOG_BACKUP	Take a log backup frequently
ACTIVE_BACKUP_OR_RESTORE	Re-evaluate the backup strategy/plan
ACTIVE_TRANSACTION	Kill erroneous or poorly written transaction
REPLICATION	Check transactional replication

# Monitoring Transaction Log Space

---

- Performance Monitor
  - Log File(s) Size (KB)
  - Log File(s) Used Size (KB)
  - Percent Log Used
  - Log Growths
- DBCC SQLPERF(LOGSPACE)
- sys.dm\_db\_log\_space\_usage (since SQL Server 2012)

# Error 9002

---

If log can't auto-grow:

- You will receive error 9002
- Rolls back uncommitted transactions
- Stops activity (writing new transactions to log file)

How to correct:

- Check the reason, then take a corresponding action
- Extend log file (if applicable)
- Add an additional log file

# DEMO

---

Excessive log growing and error 9002  
Deleting additional log files

# Log shrinking

---

## Steps:

- Run DBCC LOGININFO to estimate the number of VLFs and last active VLF
- Truncate log
  - Make log backup for Full or Bulk-logged recovery model
  - Make CHECKPOINT in Simple recovery model
- Wait for log roll-over and truncate again
- Run DBCC SHRINKFILE

# VLF Fragmentation

---

- VLF are added during log growth
  - Improper growth value leads to a big number of small or tiny VLFs
- Truncated VLFs can be in any place of transaction log
  - Leads to fragmentation in VLF sequence
- Causes problems in log activity, backups or readers
- If number of VLF is hundreds or thousands, think about VLF defragmentation

# Removing VLF Fragmentation

---

- Manually shrink file
- Repeat shrinking to reach minimum file size
- Change transaction log file size and/or autogrowth
  - VLF size should not be bigger than 500 MB
  - Perform manual growing to get optimal log size

# DEMO

---

Shrinking log file

Removing VLF fragmentation

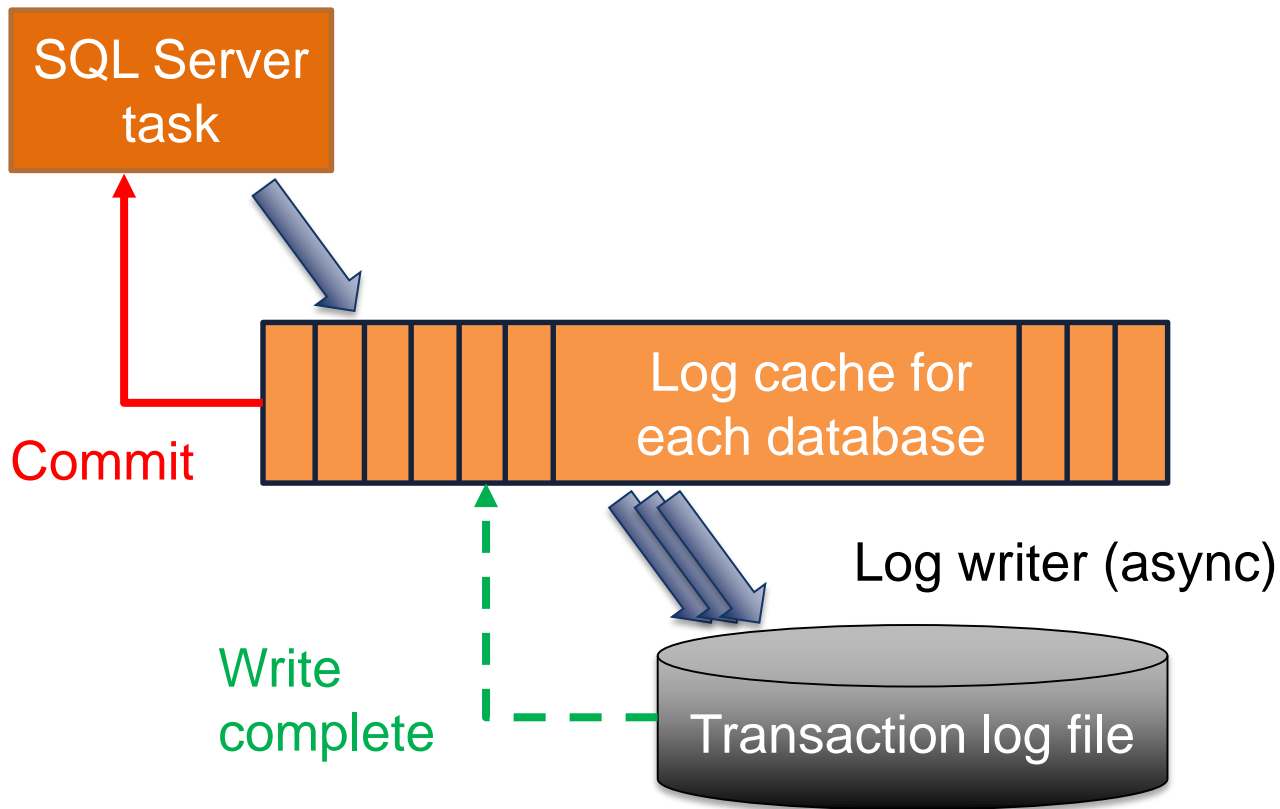


# Delayed Durability in SQL Server 2014

---

- Commit transactions BEFORE log flush
- Defined at database level
- Benefits:
  - Reducing waits
  - Increasing throughput by larger flush chunks
  - Reducing contention for log I/O
- Disadvantages:
  - Risk of data loss

# Transaction Log Flush in Delayed Durability



# DEMO

---

## Speeding up transactions with Delayed Durability

# Summary

---

- Place transaction log on separate fast physical disk
- Keep just one log file
- Monitor log size and performance
- Prevent log filling up
- Manage the number of VLFs
- Think about upgrade to the latest SQL Server versions

# Questions?



# Thanks!

