TempDB

Do This and Don’t Do That

Presented by Steve Stedman

Stedman Solutions, LLC.

http://StedmanSolutions.com

http://SteveStedman.com

About Steve

- Blogs regularly at http://SteveStedman.com
- Author SQL Server – Common Table Expressions Book
- Creator of Database Health Monitor. http://DatabaseHealth.com
- Host of the Database Corruption Challenge
- 26 years of SQL Server experience
- Freelance Database Consultant (SQL Server)
  - http://StedmanSolutions.com

Agenda

- What is TempDB, what does it do and why is that important?
  - What is in TempDB?
  - Virtual Log Files (VLFs)
- Monitoring TempDB - IO Stats, VLFs, Size, Location, Growth
- TempDB suggestions, and why
  - Multiple TempDB files
  - File growth settings
  - Trace Flags 1117, and 1118
  - TempDB on a drive of its own

Demo Environment

- SQL Server 2014 instances
  - localhost\sql2014_Clara
  - localhost\sql2014_Strax
  - localhost\sql2014_River

- Memory limit set to 1gb to simulate server constrained by memory.

What is in TempDB?

- Temporary user objects that are explicitly created, such as:
  - Global or local temporary tables
  - Temporary stored procedures
  - Table Variables
- Sorts (including DISTINCT)
- Hash (aggregates and joins)
- Spools (table and index)
- Row versions that are generated by data modification transactions in a database that uses read-committed using row versioning isolation or snapshot isolation transactions.
- Row versions that are generated by data modification transactions for features, such as: online index operations and AFTER triggers.

TempDB is Special

- No backups
- No restores
- Simple recovery only
- Cannot change owner
- Cannot drop database
- Cannot Enable CDC / Mirroring / Snapshots / Availability Groups
- If the database files are missing when SQL Server starts, the files are recreated

New Database

Select a page

- General
- Options
- Filegroups

Script ▾ Help

Database name: MyNewDatabase

Owner: <default>

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Path
MyNewDatabase	ROWS Data	PRIMARY	2	By 1 MB, Unlimited	C:\SQL2008R2Standard\MSS
MyNewDatabase_log	LOG	Not Applic...	1	By 10 percent, Unlimited	C:\SQL2008R2Standard\MSS

Add Remove

Connection

Server:
win12\standard2008r2

Connection:
sa

View connection

Progress

Ready

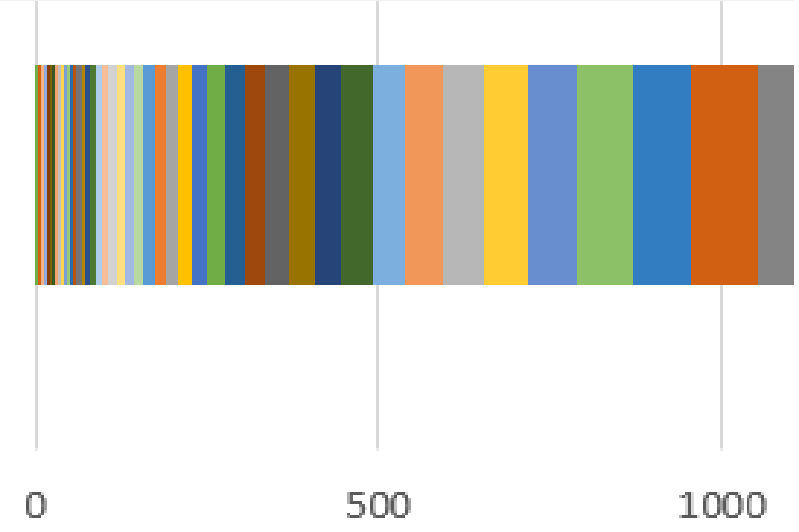
Virtual Log Files (VLFs)

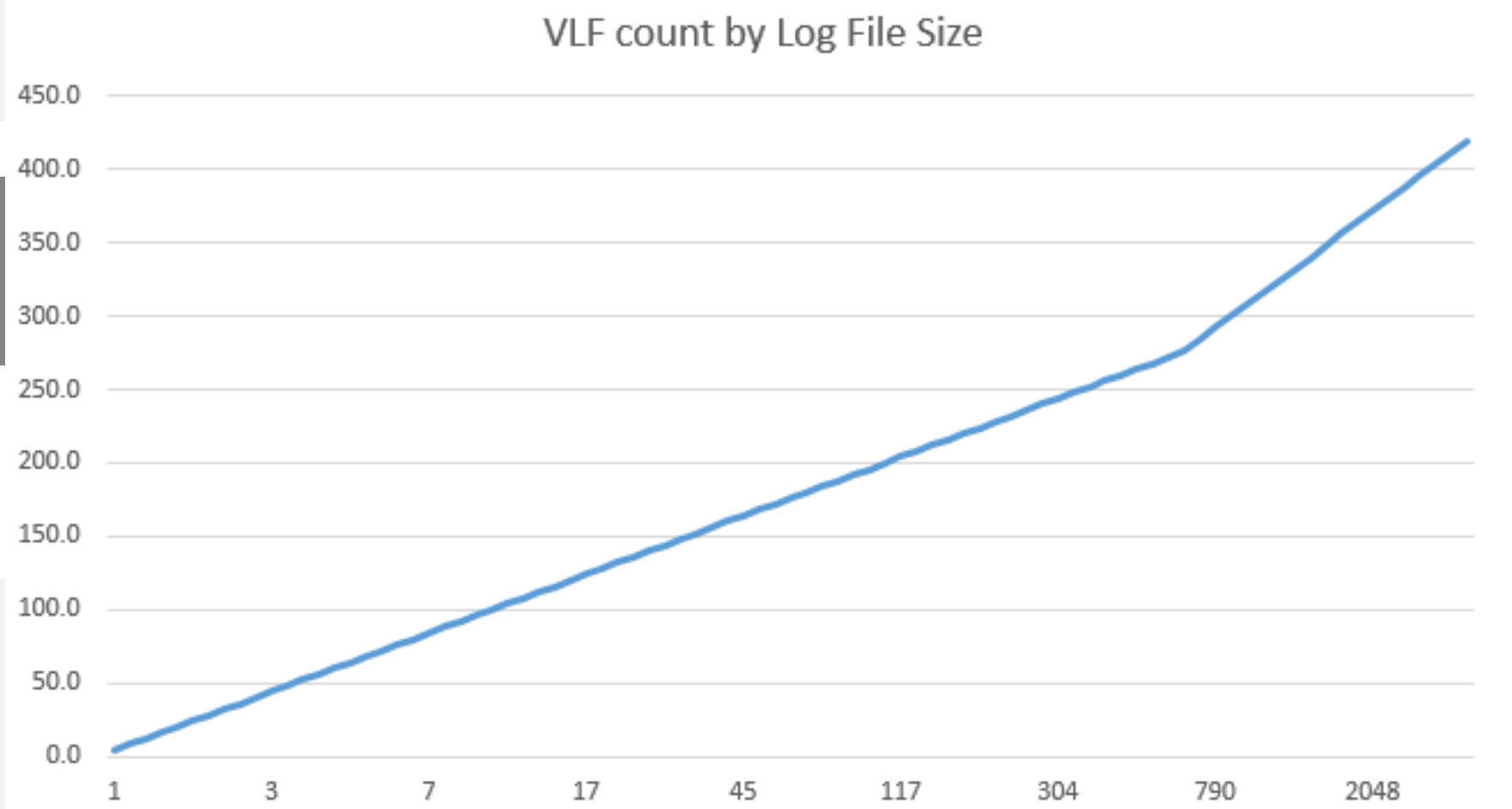
- When SQL Server allocates new log file space the files are allocated as chunks called Virtual Log Files (VLFs).
- VLFs are added in the following increments of log growth
  - up to 64mb = 4 vlfs
  - 64mb up to 1gb = 8 vlfs
  - More than 1gb = 16 vlfs
- High VLF counts (slow the system) > 500 to 1000
- To large of VLFs cause performance problems > 1gb

Log File Usage

File Growth With Default 1mb and 10% growth

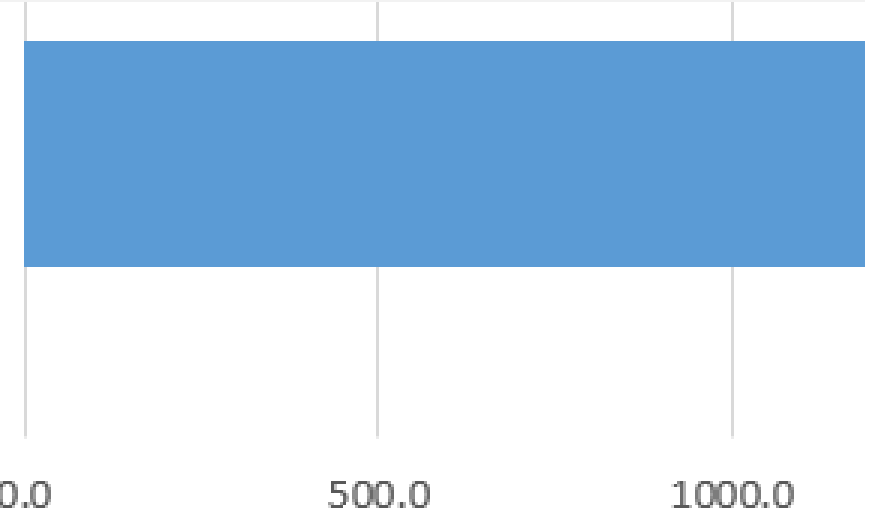
- If the log grows to 3.5gb
- VLF count will be 420
- VLF sizes vary

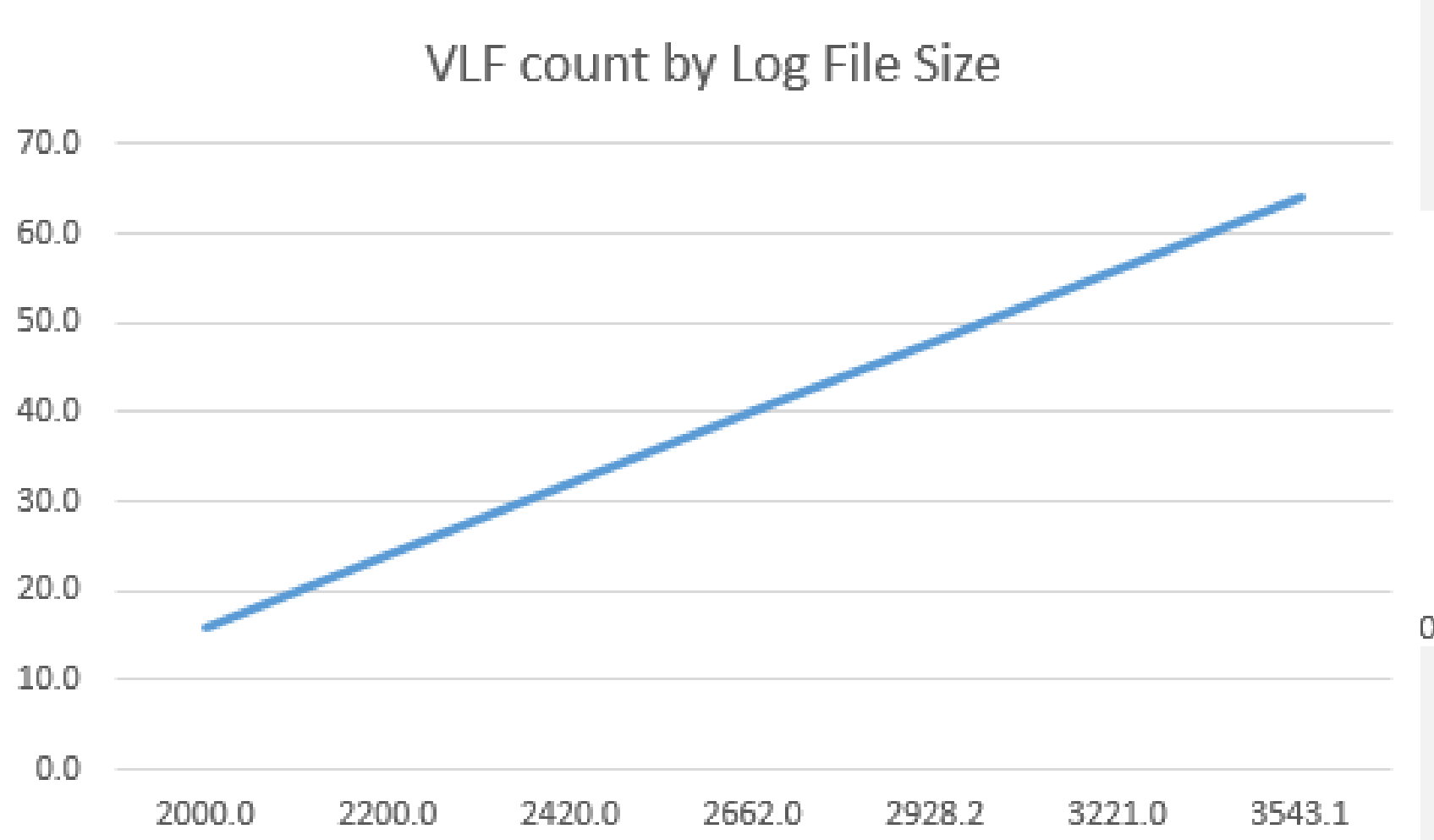




Start big, log file of 2000mb and 10% growth

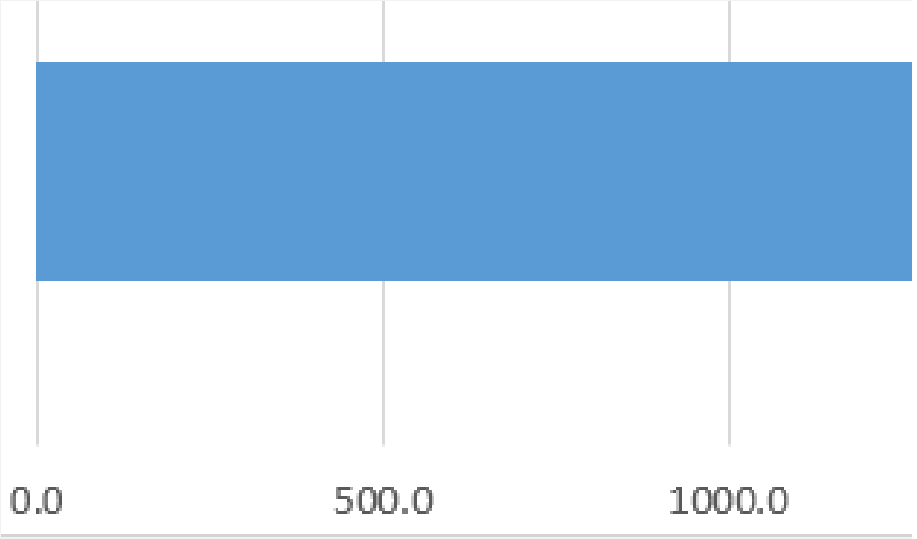
- If the log grows to 3.5gb
- VLF count will be 64
- VLF sizes up to 125mb

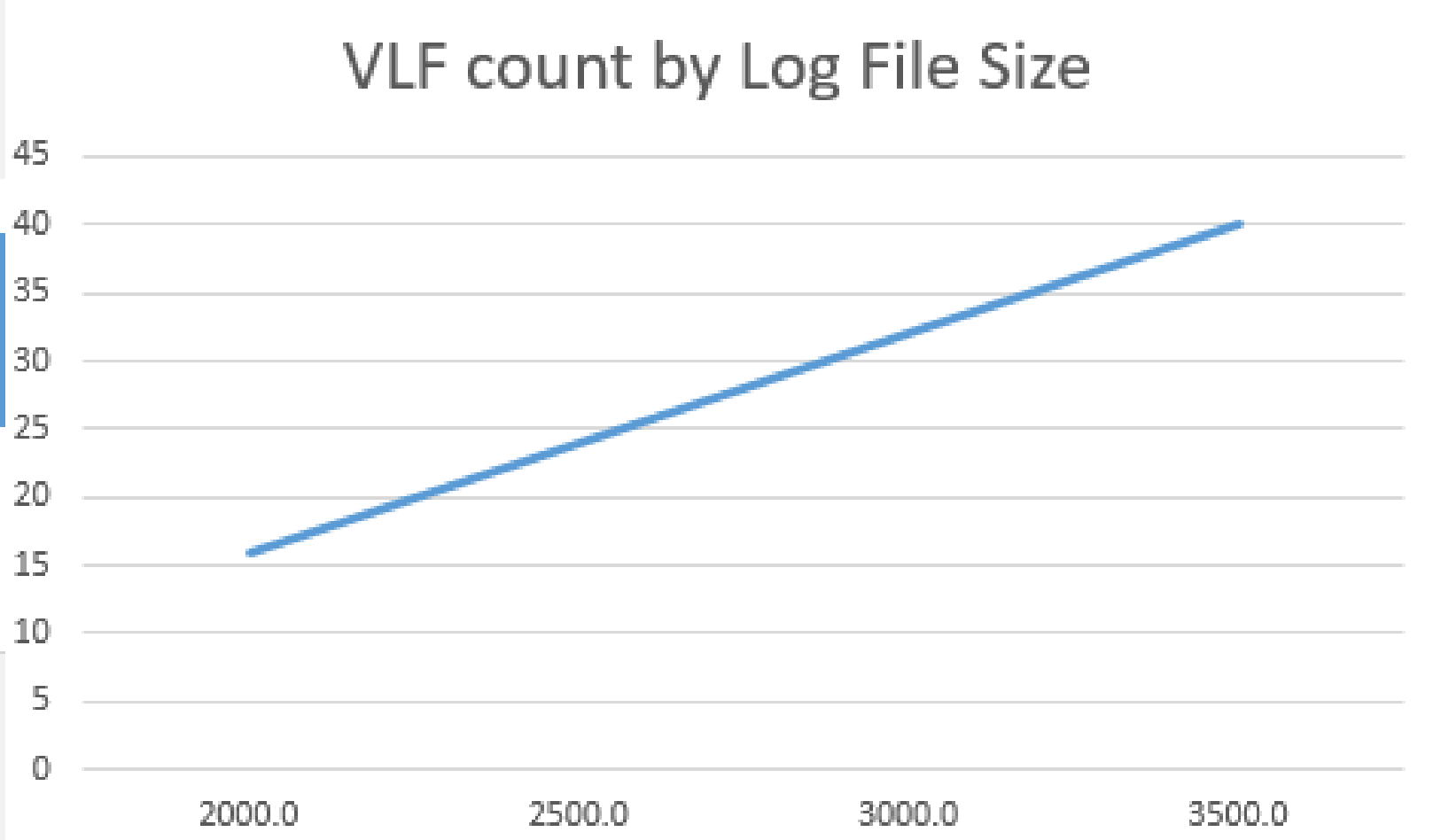




Start big, log file of 2000mb and 500mb growth

- If the log grows to 3.5gb
- VLF count will be 40
- VLF sizes 125mb to 250mb





VLF Growth Demo

DEMO

Adjusting Log Files

- Check VLF count with DBCC LOGINFO
- Adjust the settings including initial size and autogrowth
- Shrink to as small as possible (in small steps)
  - Do this off hours
- Grow to an appropriate size
- Check your autogrowth settings – confirm your VLF count

VLF Summary

- Virtual Log Files (VLFs) are part of the Transaction Log
- Too high of VLF count can negatively impact performance
- To big of VLF files can negatively impact performance
- Goldilocks: Just right
  - As large as possible
  - Keeping the VLF count under a few hundred
- Adjust as necessary as your system grows

Monitoring TempDB – IO Stats

```sql
;WITH TempDbIoStats AS (
    SELECT( 1.0 * [stats].[io_stall_write_ms] / [stats].[num_of_writes] ) AS [writeStall]
        , ( 1.0 * [stats].[io_stall_read_ms] / [stats].[num_of_reads] ) AS [readStall]
        , [stats].[file_id] AS [fileNum]
    FROM [sys].[dm_io_virtual_file_stats](2, NULL) AS [stats]
    INNER JOIN [master].[sys].[master_files] AS [files]
            ON [stats].[database_id] = [files].[database_id]
           AND [stats].file_id = [files].file_id
    WHERE [files].[type_desc] = 'ROWS'
)
SELECT 'TempDB file #' + CAST([fileNum] AS VARCHAR(10)) + ' shows slow IO (reads = ' +
        CAST(CAST([readStall] AS NUMERIC(20, 2)) AS VARCHAR(30)) + 'ms writes = ' +
   CAST(CAST([writeStall] AS NUMERIC(20, 2)) AS VARCHAR(30)) + 'ms).' as [Details]
        , 'http://DatabaseHealth.com/TempDBSlowIO' as [URL]
FROM [TempDbIoStats]
WHERE [writeStall] > 30
   OR [readStall] > 30
OPTION( RECOMPILE );
```

Monitoring TempDB - VLFs

```sql
USE TempDB;
GO
DBCC LOGINFO;
```

Monitoring TempDB - VLFs

```
use tempdb;
DECLARE @logInfoResults AS TABLE (
[RecoveryUnitId] BIGINT,  -- only on SQL Server 2012 and newer
[FileId] TINYINT,
[FileSize] BIGINT,
[StartOffset] BIGINT,
[FSeqNo] INTEGER,
[Status] TINYINT,
[Parity] TINYINT,
[CreateLSN] BIGINT );

INSERT INTO  @logInfoResults  EXEC sp_executesql N'DBCC LOGINFO WITH NO_INFOMSGS';
```

Monitoring TempDB - Size and Location

```sql
USE tempdb;
GO
SELECT size / 128.0 as sizeMB, *
  FROM sys.database_files
 ORDER by type_desc;
```

Monitoring TempDB - Growth

```
SELECT '[TempDB] data files (' + [df].[name] + ') show growth‘
FROM [tempdb].[sys].[database_files] AS [df]
INNER JOIN [tempdb].[sys].[master_files] AS [mf] ON [df].file_id = [mf].file_id
AND [mf].[database_id] = 2
WHERE [mf].[size] * 100 / [df].[size] < 70 AND [df].[type] = 0;

SELECT '[TempDB] log files (' + [df].[name] + ') show growth‘
FROM [tempdb].[sys].[database_files] AS [df]
INNER JOIN [tempdb].[sys].[master_files] AS [mf] ON [df].file_id = [mf].file_id
AND [mf].[database_id] = 2
WHERE [mf].[size] * 100 / [df].[size] < 70 AND [df].[type] = 1;
```

Server Configuration | Data Directories | TempDB | FILESTREAM

TempDB data files: tempdb.mdf, tempdb_mssql_#.ndf

Number of files: 8

Initial size (MB): 8 Total initial size (MB): 64

Autogrowth (MB): 64 Total autogrowth (MB): 512

Data directories: C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER

Add...

Remove

TempDB log file: templog.ldf

Initial size (MB): 8

Autogrowth (MB): 64

Log directory: C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER

...

Suggested: File growth settings

- Size your TempDB so that it doesn't need to grow under regular load.

Suggested: TempDB on a drive of its own

- TempDB can be the most IO intensive database on your system
- Generally put TempDB on your fastest storage.
- Even if you are in a SAN or other shared storage environment.

Suggested: Trace Flags 1117, and 1118

- **Trace flag 1117** – changes the autogrowth so that all files in a filegroup grow at the same time to the same size.
- **Trace flag 1118** - prevents contention on the SGAM pages by changing the size from 8k (single page) to 64k (8 pages).
  - **S**hared **G**lobal **A**llocation **M**ap
  - Increased from 8k to 64k

- TF1117 and TF1118 are no longer needed with SQL Server 2016. The behavior associated with these is enabled in SQL Server 2016.

Review

- What is TempDB, what does it do and why is that important?
  - What is in TempDB?
  - Virtual Log Files (VLFs)
- Monitoring TempDB - IO Stats, VLFs, Size, Location, Growth
- TempDB suggestions, and why
  - Multiple TempDB files
  - File growth settings
  - Trace Flags 1117, and 1118
  - TempDB on a drive of its own

TempDB Trivia

- True or False? TempDB prior to SQL Server is installed with 2 files.
  - tempdb.mdf
  - tempdb.ldf
- TRUE

- True or False? Those two files with their default settings can handle most production environments.
- FALSE The default configuration with only 2 files and the default file growth settings are usually inadequate for any production load.

TempDB Trivia

- TempDB should always have the same number of files as CPU cores on your system. True or False?
- FALSE This was the recommendation in SQL Server 2000, but a lot has changed since 2000. SQL Server 2005 and newer usually you can start with 8, or the number of cores up to 8, and adjust from there.

TempDB Trivia

- Table Variables use TempDB. True or False?
- TRUE Table variables use TempDB almost the same way that Temp Tables do. The only difference is Table Variables don't have statistics.

Questions?

- Any questions on anything covered?

Contact Information

- Steve Stedman
  - Email: Steve@StedmanSolutions.com
  - Blog: http://SteveStedman.com
  - Twitter: @SqlEmt
  - LinkedIn: https://www.linkedin.com/in/stevestedman

Thank you