usp_LogGrowthStats

1 Overview

This stored procedure captures **SQL Server database log information** and **performance counters**, calculates relevant metrics (size, growth, VLF info, autogrowth, backup info), **logs it into a history table** (dbo.LogGrowthHistory), and **returns the latest snapshot**.

It's useful for:

- Monitoring **log growth trends**
- Detecting **frequent log autogrowths**
- Checking **VLF fragmentation**
- Tracking **log space usage**
- Alerting for **log backups**

2 Step-by-Step Explanation

A. CTE: logs

```
WITH logs AS (
  SELECT
    DB.name AS DatabaseName,
    DB.database_id,
    MAX(DB.recovery_model_desc) AS RecoveryModel,
    SUM(CAST(MF.size AS BIGINT) * 8) AS TotalSizeKB,
    SUM(
      CASE
        WHEN MF.is_percent_growth = 0 THEN CAST(MF.growth AS BIGINT)
        ELSE CAST(MF.size AS BIGINT) * MF.growth / 100
      END * 8
    ) AS TotalGrowthKB
  FROM sys.master_files AS MF
  INNER JOIN sys.databases AS DB ON MF.database_id = DB.database_id
  WHERE MF.type = 1
  GROUP BY DB.name, DB.database_id
)
```


Purpose:

- Gathers **all log files** (MF.type = 1) for each database.
- Calculates:
  - **Total log file size** (TotalSizeKB)
  - **Total growth potential** (TotalGrowthKB)
- Uses BIGINT to avoid **arithmetic overflow** on large databases.
- Captures **recovery model** (FULL, SIMPLE, BULK_LOGGED).

B. CTE: total, growth, shrinks

total, growth, shrinks

- total: Total number of log growth events across all databases (_Total).
- growth: Log growth count per database.
- shrinks: Log shrink count per database.

Purpose:

- Helps calculate **database-specific log growth rate** as a percentage of total growths.

C. Insert into dbo.LogGrowthHistory

INSERT INTO dbo.LogGrowthHistory (...)

SELECT ...

This section:

1. Joins logs CTE with:
    - sys.databases → database metadata
    - sys.dm_db_log_stats → current log usage (active/total)
    - sys.dm_db_log_info → VLF info
    - Performance counter CTEs (growth, shrinks, total)
    - msdb.dbo.backupset → last log backup date
2. Calculates metrics:

Metric	Calculation / Source
TotalSizeMB / GB	TotalSizeKB / 1024 or / 1024^2
TotalGrowthMB / GB	TotalGrowthKB / 1024 or / 1024^2
TotalLogSizeMB	LS.total_log_size_mb from dm_db_log_stats
UsedLogSpaceMB	LS.active_log_size_mb
UsedLogSpacePercent	(active_log_size / total_log_size) * 100
FreeLogSpaceMB / Percent	total - active, 100 - percent_used
VLFCount / LargestVLFSizeMB	From sys.dm_db_log_info
GrowthRatePercent	(database growths / total growths) * 100
LogReuseWaitDesc	DB.log_reuse_wait_desc
LogBackupRequired	1 if LOG_BACKUP required
LastLogBackupDate	msdb.dbo.backupset
AutogrowthType / AutogrowthMB / MaxSizeMB	From sys.master_files, calculated safely with BIGINT

D. Return latest snapshot

SELECT *

FROM dbo.LogGrowthHistory

WHERE CaptureTime = SYSUTCDATETIME();

- After inserting, it **returns the latest captured metrics** for immediate analysis.

3 Advantages of this SP

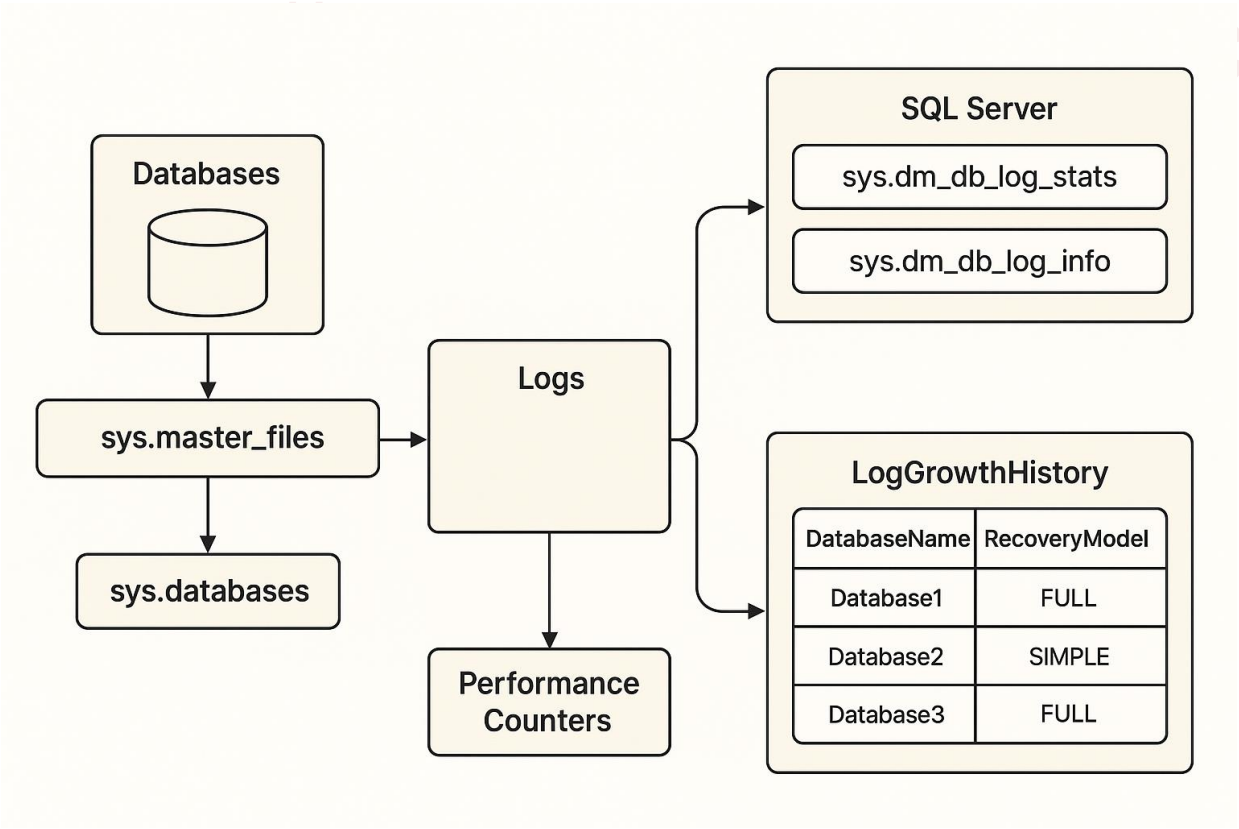
1. **Centralized Log Monitoring**
    - Tracks **all SQL Server databases** in one table.
    - Historical trends help with **capacity planning** and **performance tuning**.
2. **Prevents Arithmetic Overflow**
    - Uses BIGINT for all large log and growth calculations.
    - Works safely even for **multi-terabyte log files**.
3. **Detailed Metrics**

- Total size, growth, VLF info, used/free space.
- Autogrowth type and size.
- Log reuse wait reason.
- Last log backup info.

4. **Supports Alerting & Automation**
   - Growth rate percentage can trigger **alerts** for frequent autogrowths.
   - Log backup required flag identifies **databases needing attention**.
5. **VLF Awareness**
   - Helps detect **VLF fragmentation**, which impacts log performance.
6. **Historical Logging**
   - Stores metrics in LogGrowthHistory for **trend analysis over time**.
7. **Safe for Large Databases**
   - Casting to BIGINT ensures **no arithmetic overflow errors**.

4 Use Cases

- Monitoring **log file growth trends** in production.
- Identifying **databases with frequent autogrowths**.
- Planning **log backup strategy**.
- Detecting **VLF fragmentation issues** early.
- Performing **historical analysis** to optimize log sizes.



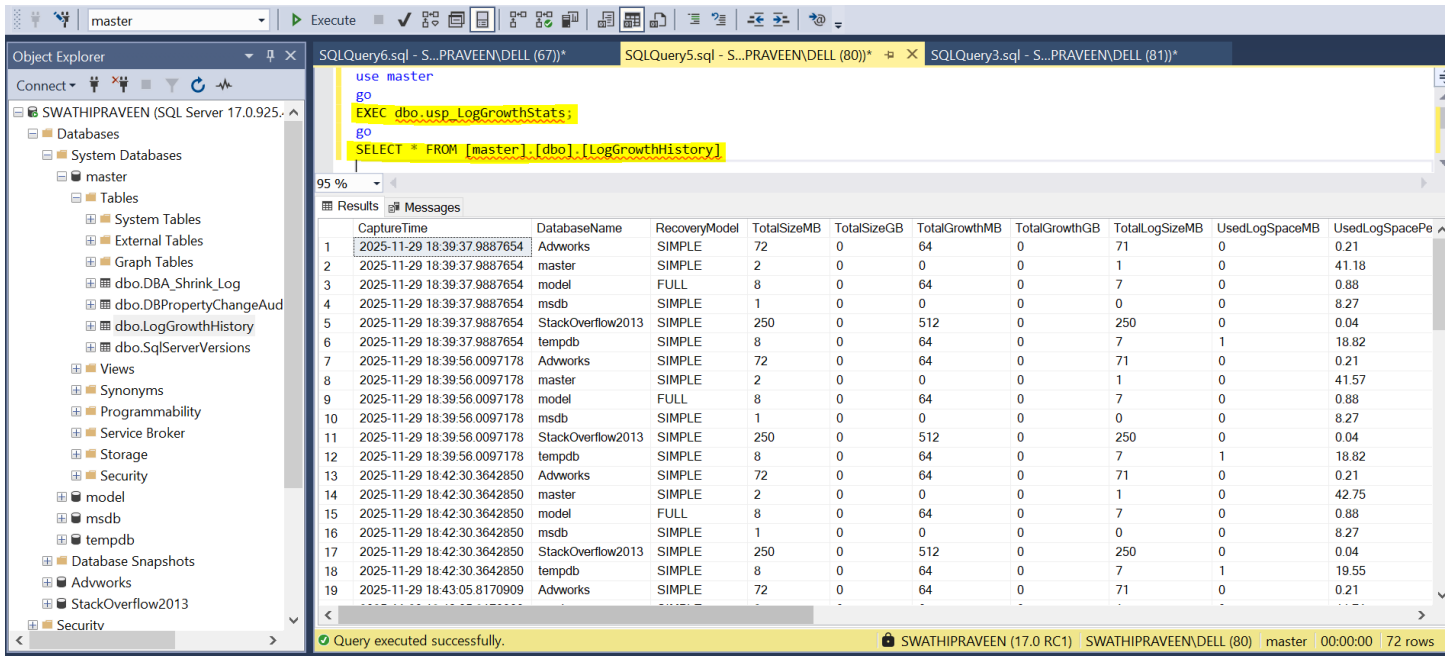
```
Use master
Go
IF OBJECT_ID('dbo.usp_LogGrowthStats', 'P') IS NOT NULL
   DROP PROCEDURE dbo.usp_LogGrowthStats;
GO

CREATE PROCEDURE dbo.usp_LogGrowthStats
AS
BEGIN
   SET NOCOUNT ON;

   ----------------------------------------------------------------
   -- CTEs for database log stats and performance counters
   ----------------------------------------------------------------
   WITH logs AS (
      SELECT
         DB.name AS DatabaseName,
         DB.database_id,
         MAX(DB.recovery_model_desc) AS RecoveryModel,
         SUM(CAST(MF.size AS BIGINT) * 8) AS TotalSizeKB,
         SUM(
            CASE
               WHEN MF.is_percent_growth = 0 THEN CAST(MF.growth AS BIGINT)
               ELSE CAST(MF.size AS BIGINT) * MF.growth / 100
            END * 8
         ) AS TotalGrowthKB
      FROM sys.master_files AS MF
      INNER JOIN sys.databases AS DB ON MF.database_id = DB.database_id
      WHERE MF.type = 1
      GROUP BY DB.name, DB.database_id
   ),
   total AS (
      SELECT OPC.cntr_value AS TotalCounter
      FROM sys.dm_os_performance_counters AS OPC
      WHERE OPC.object_name LIKE N'%SQL%:Databases%'
        AND OPC.counter_name = N'Log Growths'
        AND OPC.instance_name = N'_Total'
   ),
   growth AS (
      SELECT OPC.instance_name AS DatabaseName,
            OPC.cntr_value AS Growths
      FROM sys.dm_os_performance_counters AS OPC
      WHERE OPC.object_name LIKE N'%SQL%:Databases%'
        AND OPC.counter_name = N'Log Growths'
        AND OPC.instance_name <> N'_Total'
   ),
   shrinks AS (
      SELECT OPC.instance_name AS DatabaseName,
            OPC.cntr_value AS Shrinks
      FROM sys.dm_os_performance_counters AS OPC
```

```
    WHERE OPC.object_name LIKE N'%SQL%:Databases%'
      AND OPC.counter_name = N'Log Shrinks'
      AND OPC.instance_name <> N'_Total'
)

-------------------------------------------------------------------
-- Insert snapshot into history table
-------------------------------------------------------------------
INSERT INTO dbo.LogGrowthHistory
(
    DatabaseName,
    RecoveryModel,
    TotalSizeMB,
    TotalSizeGB,
    TotalGrowthMB,
    TotalGrowthGB,
    TotalLogSizeMB,
    UsedLogSpaceMB,
    UsedLogSpacePercent,
    FreeLogSpaceMB,
    FreeLogSpacePercent,
    VLFCount,
    LargestVLFSizeMB,
    Growths,
    Shrinks,
    GrowthRatePercent,
    LogReuseWaitDesc,
    LogBackupRequired,
    LastLogBackupDate,
    AutogrowthType,
    AutogrowthMB,
    MaxSizeMB
)
SELECT
    L.DatabaseName,
    L.RecoveryModel,

    -- Total size/growth as BIGINT
    CONVERT(BIGINT, L.TotalSizeKB / 1024.0) AS TotalSizeMB,
    CONVERT(BIGINT, L.TotalSizeKB / 1024.0 / 1024.0) AS TotalSizeGB,
    CONVERT(BIGINT, L.TotalGrowthKB / 1024.0) AS TotalGrowthMB,
    CONVERT(BIGINT, L.TotalGrowthKB / 1024.0 / 1024.0) AS TotalGrowthGB,

    -- Log space from dm_db_log_stats
    CONVERT(BIGINT, LS.total_log_size_mb) AS TotalLogSizeMB,
    CONVERT(BIGINT, LS.active_log_size_mb) AS UsedLogSpaceMB,
    CONVERT(DECIMAL(10,2), (LS.active_log_size_mb / LS.total_log_size_mb) * 100.0) AS UsedLogSpacePercent,
    CONVERT(BIGINT, LS.total_log_size_mb - LS.active_log_size_mb) AS FreeLogSpaceMB,
    CONVERT(DECIMAL(10,2), 100.0 - ((LS.active_log_size_mb / LS.total_log_size_mb) * 100.0)) AS FreeLogSpacePercent,

    -- VLF info
```

```
    VLF.VLFCount,
    CONVERT(BIGINT, VLF.LargestVLFSizeMB) AS LargestVLFSizeMB,

    -- Growth / Shrink counters
    G.Growths,
    S.Shrinks,

    -- Growth rate %
    CONVERT(DECIMAL(10,2),
      CASE WHEN T.TotalCounter = 0 THEN 0.0 ELSE 100.0 * G.Growths / T.TotalCounter END
    ) AS GrowthRatePercent,

    -- Log reuse wait & log backup info
    DB.log_reuse_wait_desc,
    CASE WHEN DB.log_reuse_wait_desc = 'LOG_BACKUP' THEN 1 ELSE 0 END AS LogBackupRequired,
    LB.LastLogBackupDate,

    -- Autogrowth info (log file)
    CASE
      WHEN MF.is_percent_growth = 1 THEN 'Percent'
      ELSE 'MB'
    END AS AutogrowthType,
    CASE
      WHEN MF.is_percent_growth = 1 THEN CONVERT(BIGINT, CAST(MF.size AS BIGINT) * MF.growth / 100 * 8 / 1024.0)
      ELSE CONVERT(BIGINT, CAST(MF.growth AS BIGINT) * 8 / 1024.0)
    END AS AutogrowthMB,
    CASE
      WHEN MF.max_size = -1 THEN -1
      ELSE CONVERT(BIGINT, CAST(MF.max_size AS BIGINT) * 8 / 1024.0)
    END AS MaxSizeMB

FROM logs L
INNER JOIN sys.databases DB ON L.database_id = DB.database_id
CROSS APPLY sys.dm_db_log_stats(L.database_id) AS LS
CROSS APPLY (
    SELECT COUNT(*) AS VLFCount,
           MAX(vlf_size_mb) AS LargestVLFSizeMB
    FROM sys.dm_db_log_info(L.database_id)
) AS VLF
LEFT JOIN growth G ON L.DatabaseName = G.DatabaseName
LEFT JOIN shrinks S ON L.DatabaseName = S.DatabaseName
CROSS JOIN total T
INNER JOIN sys.master_files MF
    ON L.database_id = MF.database_id AND MF.type = 1
OUTER APPLY (
    SELECT MAX(backup_finish_date) AS LastLogBackupDate
    FROM msdb.dbo.backupset
    WHERE database_name = L.DatabaseName
      AND type = 'L'
) AS LB;
```

```
    ------------------------------------------------------------
    -- Return latest snapshot immediately
    ------------------------------------------------------------
    SELECT *
    FROM dbo.LogGrowthHistory
    WHERE CaptureTime = SYSUTCDATETIME();
END
GO
```

```
use master
go
EXEC dbo.usp_LogGrowthStats;
-- No results are generated when this SP is executed. Instead, the output is saved in below mentioned table.
go
SELECT * FROM [master].[dbo].[LogGrowthHistory]
```

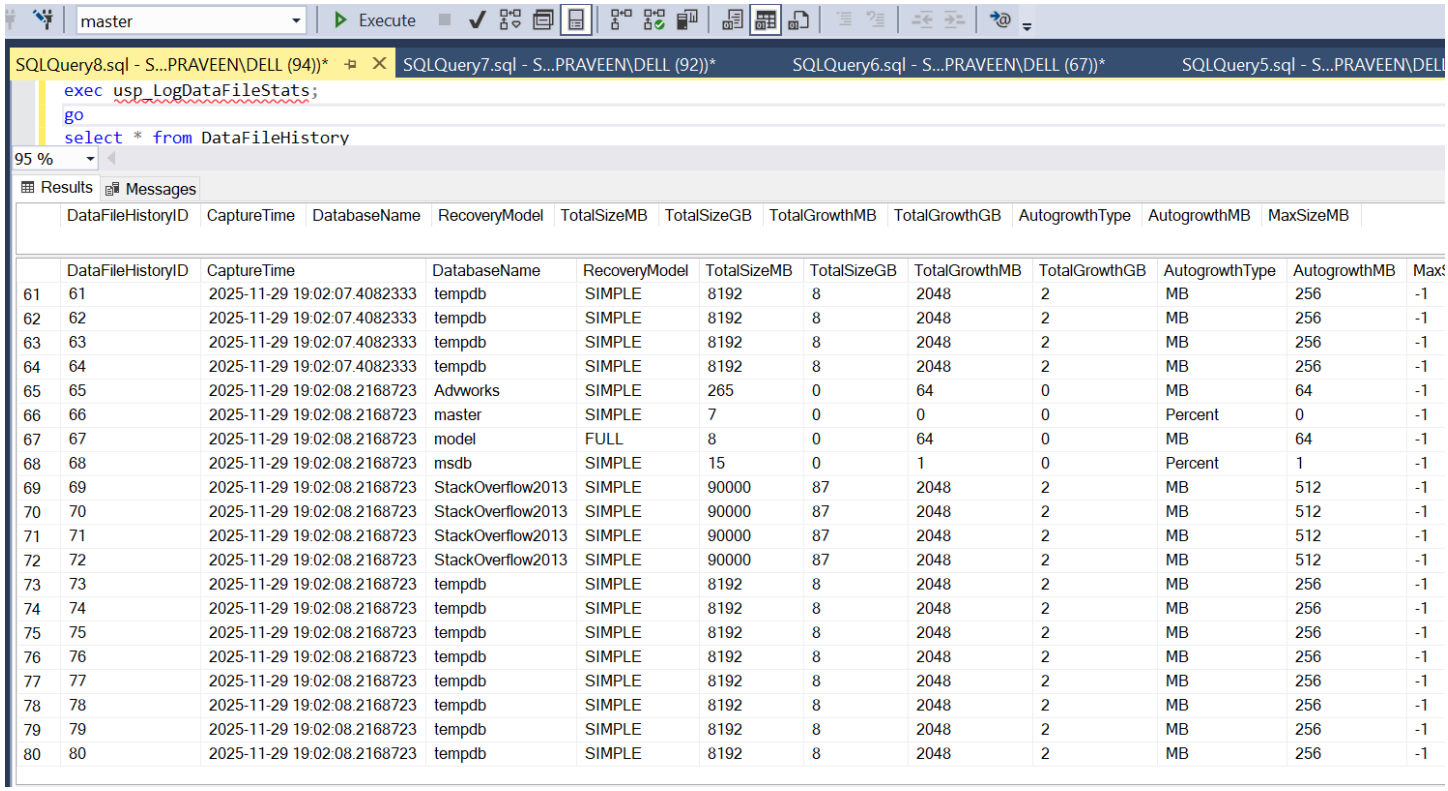


UsedLogSpacePercent	FreeLogSpaceMB	FreeLogSpacePercent	VLFCount	LargestVLF...	Growths	Shrinks	Growth...	LogReuseWaitDesc	LogBacku...	Last...	AutogrowthType	AutogrowthMB	MaxSizeMB
0.21	71	99.79	8	16	0	0	0.00	NOTHING	0	NU...	MB	64	2097152
41.18	1	58.82	8	0	0	0	0.00	NOTHING	0	NU...	Percent	0	-1
0.88	7	99.12	3	7	0	0	0.00	NOTHING	0	NU...	MB	64	-1
8.27	0	91.73	4	0	0	0	0.00	NOTHING	0	NU...	Percent	0	2097152
0.04	250	99.96	3	124	0	0	0.00	NOTHING	0	NU...	MB	512	2097152
18.82	6	81.18	4	2	0	0	0.00	NOTHING	0	NU...	MB	64	-1
0.21	71	99.79	8	16	0	0	0.00	NOTHING	0	NU...	MB	64	2097152
41.57	1	58.43	8	0	0	0	0.00	NOTHING	0	NU...	Percent	0	-1
0.88	7	99.12	3	7	0	0	0.00	NOTHING	0	NU...	MB	64	-1
8.27	0	91.73	4	0	0	0	0.00	NOTHING	0	NU...	Percent	0	2097152
0.04	250	99.96	3	124	0	0	0.00	NOTHING	0	NU...	MB	512	2097152
18.82	6	81.18	4	2	0	0	0.00	NOTHING	0	NU...	MB	64	-1
0.21	71	99.79	8	16	0	0	0.00	NOTHING	0	NU...	MB	64	2097152
42.75	1	57.25	8	0	0	0	0.00	NOTHING	0	NU...	Percent	0	-1
0.88	7	99.12	3	7	0	0	0.00	NOTHING	0	NU...	MB	64	-1
8.27	0	91.73	4	0	0	0	0.00	NOTHING	0	NU...	Percent	0	2097152
0.04	250	99.96	3	124	0	0	0.00	NOTHING	0	NU...	MB	512	2097152
19.55	6	80.45	4	2	0	0	0.00	NOTHING	0	NU...	MB	64	-1
0.21	71	99.79	8	16	0	0	0.00	NOTHING	0	NU...	MB	64	2097152

usp_LogDataFileStats → This is used to audit Data File Size Space details

```
use master
go
CREATE TABLE dbo.DataFileHistory
(
   DataFileHistoryID BIGINT IDENTITY(1,1) PRIMARY KEY,
   CaptureTime DATETIME2 DEFAULT SYSUTCDATETIME(),
   DatabaseName SYSNAME NOT NULL,
   RecoveryModel NVARCHAR(60),
   TotalSizeMB BIGINT,
   TotalSizeGB BIGINT,
   TotalGrowthMB BIGINT,
   TotalGrowthGB BIGINT,
   AutogrowthType NVARCHAR(20),
   AutogrowthMB BIGINT,
   MaxSizeMB BIGINT
);

Use master
go
IF OBJECT_ID('dbo.usp_LogDataFileStats', 'P') IS NOT NULL
   DROP PROCEDURE dbo.usp_LogDataFileStats;
GO
CREATE PROCEDURE dbo.usp_LogDataFileStats
AS
BEGIN
   SET NOCOUNT ON;

   ------------------------------------------------------------
   -- CTE for database data file stats
   ------------------------------------------------------------
   WITH datafiles AS (
      SELECT
         DB.name AS DatabaseName,
         DB.database_id,
         MAX(DB.recovery_model_desc) AS RecoveryModel,
         SUM(CAST(MF.size AS BIGINT) * 8) AS TotalSizeKB,
         SUM(
            CASE
               WHEN MF.is_percent_growth = 0 THEN CAST(MF.growth AS BIGINT)
               ELSE CAST(MF.size AS BIGINT) * MF.growth / 100
            END * 8
         ) AS TotalGrowthKB
      FROM sys.master_files AS MF
      INNER JOIN sys.databases AS DB ON MF.database_id = DB.database_id
      WHERE MF.type = 0  -- Data files only
      GROUP BY DB.name, DB.database_id
```

```
    )

    ---------------------------------------------------------------
    -- Insert snapshot into history table
    ---------------------------------------------------------------
    INSERT INTO dbo.DataFileHistory
    (
       DatabaseName,
       RecoveryModel,
       TotalSizeMB,
       TotalSizeGB,
       TotalGrowthMB,
       TotalGrowthGB,
       AutogrowthType,
       AutogrowthMB,
       MaxSizeMB
    )
    SELECT
       DF.DatabaseName,
       DF.RecoveryModel,
       CONVERT(BIGINT, DF.TotalSizeKB / 1024.0) AS TotalSizeMB,
       CONVERT(BIGINT, DF.TotalSizeKB / 1024.0 / 1024.0) AS TotalSizeGB,
       CONVERT(BIGINT, DF.TotalGrowthKB / 1024.0) AS TotalGrowthMB,
       CONVERT(BIGINT, DF.TotalGrowthKB / 1024.0 / 1024.0) AS TotalGrowthGB,
       CASE
          WHEN MF.is_percent_growth = 1 THEN 'Percent'
          ELSE 'MB'
       END AS AutogrowthType,
       CASE
          WHEN MF.is_percent_growth = 1 THEN CONVERT(BIGINT, CAST(MF.size AS BIGINT) * MF.growth / 100 * 8 / 1024.0)
          ELSE CONVERT(BIGINT, CAST(MF.growth AS BIGINT) * 8 / 1024.0)
       END AS AutogrowthMB,
       CASE
          WHEN MF.max_size = -1 THEN -1
          ELSE CONVERT(BIGINT, CAST(MF.max_size AS BIGINT) * 8 / 1024.0)
       END AS MaxSizeMB
    FROM datafiles DF
    INNER JOIN sys.master_files MF
       ON DF.database_id = MF.database_id AND MF.type = 0;

    ---------------------------------------------------------------
    -- Return latest snapshot immediately
    ---------------------------------------------------------------
    SELECT *
    FROM dbo.DataFileHistory
    WHERE CaptureTime = SYSUTCDATETIME();
END
GO
```



	DataFileHistoryID	CaptureTime	DatabaseName	RecoveryModel	TotalSizeMB	TotalSizeGB	TotalGrowthMB	TotalGrowthGB	AutogrowthType	AutogrowthMB	MaxSizeMB
61	61	2025-11-29 19:02:07.4082333	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
62	62	2025-11-29 19:02:07.4082333	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
63	63	2025-11-29 19:02:07.4082333	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
64	64	2025-11-29 19:02:07.4082333	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
65	65	2025-11-29 19:02:08.2168723	Advworks	SIMPLE	265	0	64	0	MB	64	-1
66	66	2025-11-29 19:02:08.2168723	master	SIMPLE	7	0	0	0	Percent	0	-1
67	67	2025-11-29 19:02:08.2168723	model	FULL	8	0	64	0	MB	64	-1
68	68	2025-11-29 19:02:08.2168723	msdb	SIMPLE	15	0	1	0	Percent	1	-1
69	69	2025-11-29 19:02:08.2168723	StackOverflow2013	SIMPLE	90000	87	2048	2	MB	512	-1
70	70	2025-11-29 19:02:08.2168723	StackOverflow2013	SIMPLE	90000	87	2048	2	MB	512	-1
71	71	2025-11-29 19:02:08.2168723	StackOverflow2013	SIMPLE	90000	87	2048	2	MB	512	-1
72	72	2025-11-29 19:02:08.2168723	StackOverflow2013	SIMPLE	90000	87	2048	2	MB	512	-1
73	73	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
74	74	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
75	75	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
76	76	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
77	77	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
78	78	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
79	79	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1
80	80	2025-11-29 19:02:08.2168723	tempdb	SIMPLE	8192	8	2048	2	MB	256	-1

usp_LogAndDataFileStats → Captures both Data and Log File Size details

```
IF OBJECT_ID('dbo.usp_LogAndDataFileStats', 'P') IS NOT NULL
   DROP PROCEDURE dbo.usp_LogAndDataFileStats;
GO

-- Ensure history table exists
IF OBJECT_ID('dbo.FileGrowthHistory', 'U') IS NULL
BEGIN
   CREATE TABLE dbo.FileGrowthHistory
   (
      CaptureTime DATETIME2 NOT NULL,
      DatabaseName SYSNAME NOT NULL,
      RecoveryModel NVARCHAR(60),

      -- Data file info
      DataFileTotalSizeMB BIGINT,
      DataFileTotalSizeGB BIGINT,
      DataFileTotalGrowthMB BIGINT,
      DataFileTotalGrowthGB BIGINT,
      DataFileAutogrowthType NVARCHAR(20),
      DataFileAutogrowthMB BIGINT,
      DataFileMaxSizeMB BIGINT,

      -- Log file info
      LogFileTotalSizeMB BIGINT,
      LogFileUsedMB BIGINT,
      LogFileFreeMB BIGINT,
      LogFileUsedPercent DECIMAL(10,2),
      LogFileFreePercent DECIMAL(10,2),
      VLFCount INT,
      LargestVLFSizeMB BIGINT,
      LogGrowths BIGINT,
      LogShrinks BIGINT,
      LogGrowthRatePercent DECIMAL(10,2),
      LogReuseWaitDesc NVARCHAR(60),
      LogBackupRequired BIT,
      LastLogBackupDate DATETIME2,
      LogAutogrowthType NVARCHAR(20),
      LogAutogrowthMB BIGINT,
      LogMaxSizeMB BIGINT,

      PRIMARY KEY (CaptureTime, DatabaseName)
   );
END
GO

CREATE PROCEDURE dbo.usp_LogAndDataFileStats
AS
```

```
BEGIN
  SET NOCOUNT ON;

  DECLARE @CaptureTime DATETIME2 = SYSUTCDATETIME();

  ----------------------------------------------------------------
  -- Aggregate Data File Info
  ----------------------------------------------------------------
  ;WITH DataFiles AS (
    SELECT
      DB.database_id,
      DB.name AS DatabaseName,
      DB.recovery_model_desc AS RecoveryModel,
      SUM(CAST(MF.size AS BIGINT) * 8) AS TotalSizeKB,
      SUM(
        CASE
          WHEN MF.is_percent_growth = 0 THEN CAST(MF.growth AS BIGINT)
          ELSE CAST(MF.size AS BIGINT) * MF.growth / 100
        END * 8
      ) AS TotalGrowthKB,
      MAX(CAST(MF.growth AS BIGINT)) AS MaxGrowth,
      MAX(CAST(MF.is_percent_growth AS INT)) AS IsPercentGrowth,
      MAX(CAST(MF.max_size AS BIGINT)) AS MaxSize
    FROM sys.master_files MF
    INNER JOIN sys.databases DB ON MF.database_id = DB.database_id
    WHERE MF.type = 0 -- Data files
    GROUP BY DB.database_id, DB.name, DB.recovery_model_desc
  ),
  ----------------------------------------------------------------
  -- Aggregate Log File Info per Database
  ----------------------------------------------------------------
  LogFiles AS (
    SELECT
      DB.database_id,
      DB.name AS DatabaseName,
      SUM(CAST(MF.size AS BIGINT) * 8) AS TotalSizeKB,
      LS.total_log_size_mb,
      LS.active_log_size_mb,
      VLFInfo.VLFCount,
      VLFInfo.LargestVLFSizeMB,
      G.Growths,
      S.Shrinks,
      DB.log_reuse_wait_desc,
      LB.LastLogBackupDate,
      MAX(CAST(MF.growth AS BIGINT)) AS MaxGrowth,
      MAX(CAST(MF.is_percent_growth AS INT)) AS IsPercentGrowth,
      MAX(CAST(MF.max_size AS BIGINT)) AS MaxSize
    FROM sys.databases DB
    INNER JOIN sys.master_files MF ON DB.database_id = MF.database_id AND MF.type = 1
    CROSS APPLY sys.dm_db_log_stats(DB.database_id) AS LS
    CROSS APPLY (
```

```
        SELECT COUNT(*) AS VLFCount, MAX(vlf_size_mb) AS LargestVLFSizeMB
        FROM sys.dm_db_log_info(DB.database_id)
    ) AS VLFInfo
    LEFT JOIN (
        SELECT instance_name AS DatabaseName, cntr_value AS Growths
        FROM sys.dm_os_performance_counters
        WHERE counter_name = 'Log Growths' AND instance_name <> '_Total'
    ) G ON G.DatabaseName = DB.name
    LEFT JOIN (
        SELECT instance_name AS DatabaseName, cntr_value AS Shrinks
        FROM sys.dm_os_performance_counters
        WHERE counter_name = 'Log Shrinks' AND instance_name <> '_Total'
    ) S ON S.DatabaseName = DB.name
    OUTER APPLY (
        SELECT MAX(backup_finish_date) AS LastLogBackupDate
        FROM msdb.dbo.backupset
        WHERE database_name = DB.name AND type = 'L'
    ) AS LB
    GROUP BY DB.database_id, DB.name, LS.total_log_size_mb, LS.active_log_size_mb,
             VLFInfo.VLFCount, VLFInfo.LargestVLFSizeMB, G.Growths, S.Shrinks,
             DB.log_reuse_wait_desc, LB.LastLogBackupDate
)

------------------------------------------------------------------
-- Insert snapshot (one row per database)
------------------------------------------------------------------
INSERT INTO dbo.FileGrowthHistory
(
    CaptureTime,
    DatabaseName,
    RecoveryModel,
    DataFileTotalSizeMB,
    DataFileTotalSizeGB,
    DataFileTotalGrowthMB,
    DataFileTotalGrowthGB,
    DataFileAutogrowthType,
    DataFileAutogrowthMB,
    DataFileMaxSizeMB,
    LogFileTotalSizeMB,
    LogFileUsedMB,
    LogFileFreeMB,
    LogFileUsedPercent,
    LogFileFreePercent,
    VLFCount,
    LargestVLFSizeMB,
    LogGrowths,
    LogShrinks,
    LogReuseWaitDesc,
    LogBackupRequired,
    LastLogBackupDate,
    LogAutogrowthType,
```

```
        LogAutogrowthMB,
        LogMaxSizeMB
    )
    SELECT
        @CaptureTime,
        DF.DatabaseName,
        DF.RecoveryModel,
        CONVERT(BIGINT, DF.TotalSizeKB / 1024.0),
        CONVERT(BIGINT, DF.TotalSizeKB / 1024.0 / 1024.0),
        CONVERT(BIGINT, DF.TotalGrowthKB / 1024.0),
        CONVERT(BIGINT, DF.TotalGrowthKB / 1024.0 / 1024.0),
        CASE WHEN DF.IsPercentGrowth = 1 THEN 'Percent' ELSE 'MB' END,
        CASE WHEN DF.IsPercentGrowth = 1 THEN CONVERT(BIGINT, DF.TotalSizeKB * DF.MaxGrowth / 100 / 1024.0)
            ELSE CONVERT(BIGINT, DF.MaxGrowth / 1024.0) END,
        CASE WHEN DF.MaxSize = -1 THEN -1 ELSE CONVERT(BIGINT, DF.MaxSize / 1024.0) END,
        LF.TotalSizeKB / 1024,
        LF.active_log_size_mb,
        LF.total_log_size_mb - LF.active_log_size_mb,
        CONVERT(DECIMAL(10,2), (LF.active_log_size_mb / NULLIF(LF.total_log_size_mb,0))*100.0),
        CONVERT(DECIMAL(10,2), 100 - ((LF.active_log_size_mb / NULLIF(LF.total_log_size_mb,0))*100.0)),
        LF.VLFCount,
        LF.LargestVLFSizeMB,
        LF.Growths,
        LF.Shrinks,
        LF.log_reuse_wait_desc,
        CASE WHEN LF.log_reuse_wait_desc='LOG_BACKUP' THEN 1 ELSE 0 END,
        LF.LastLogBackupDate,
        CASE WHEN LF.IsPercentGrowth = 1 THEN 'Percent' ELSE 'MB' END,
        CASE WHEN LF.IsPercentGrowth = 1 THEN CONVERT(BIGINT, LF.TotalSizeKB * LF.MaxGrowth / 100 / 1024.0)
            ELSE CONVERT(BIGINT, LF.MaxGrowth / 1024.0) END,
        CASE WHEN LF.MaxSize=-1 THEN -1 ELSE CONVERT(BIGINT, LF.MaxSize / 1024.0) END
    FROM DataFiles DF
    INNER JOIN LogFiles LF ON DF.database_id = LF.database_id;

    ----------------------------------------------------------------
    -- Return latest snapshot
    ----------------------------------------------------------------
    SELECT *
    FROM dbo.FileGrowthHistory
    WHERE CaptureTime = @CaptureTime;
END
GO
```

```
exec usp_LogDataFileStats
go
select * from FileGrowthHistory
```