TempDB Improvements

John Deardurff

What does this session cover?

What is the TempDB?

Types of TempDB Contention

Optimizing the TempDB Database

TempDB Performance Improvements

What is the TempDB database?

What is the TempDB database?

System database

- Available to all users with the same structure as user databases.
- Operations are minimally logged.
- Re-created every time SQL Server is started.

Workload

- Used for temporary (non-durable) storage.
- Object and data frequently being created and destroyed.
- Very high concurrency.
- Backup and restore operations are not allowed on TempDB.

What is stored in TempDB?

Temporary user objects

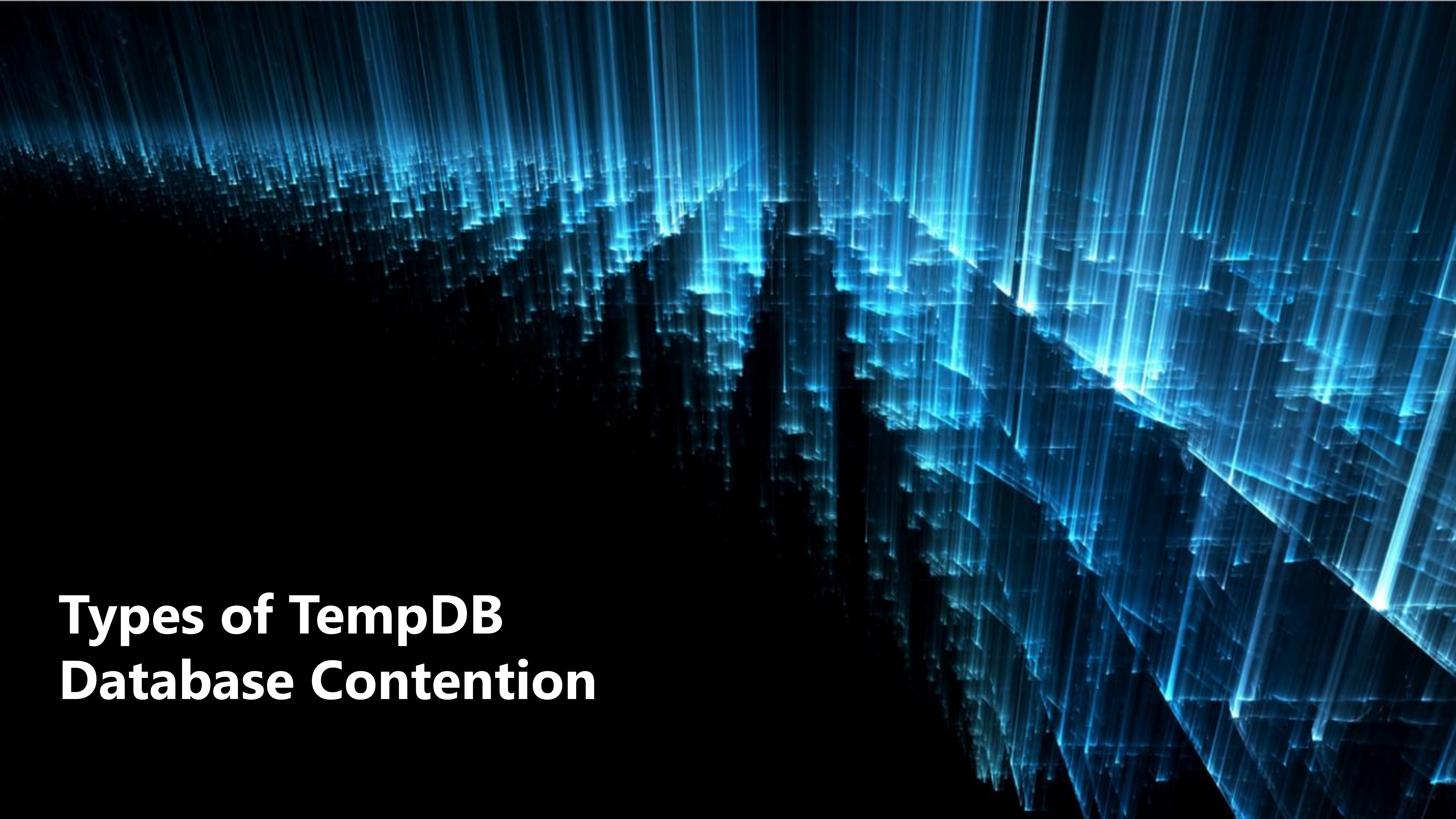
- Global or local temporary tables and indexes
- Temporary stored procedures
- Table variables
- Tables returned in table-valued functions

Internal objects

- Worktables to store intermediate results for spools, cursors, sorts, and temporary LOB storage.
- Work files for hash join or hash aggregate operations.

Version stores

- Common row version store and online-index-build version store
- Version stores can be moved to user databases by enabling Accelerated Database Recovery (ADR) in SQL Server 2019



Types of TempDB Database Contention

Types of TempDB Contention

Object allocation contention

- The wait type is PAGELATCH_UP on pages in TempDB.
- These pages might be PFS (2:1:1) or SGAM (2:1:3) pages in TempDB.

Metadata contention

- The wait type is PAGELATCH_EX on **sysschobjs** in TempDB.

Temp Table Cache Contention

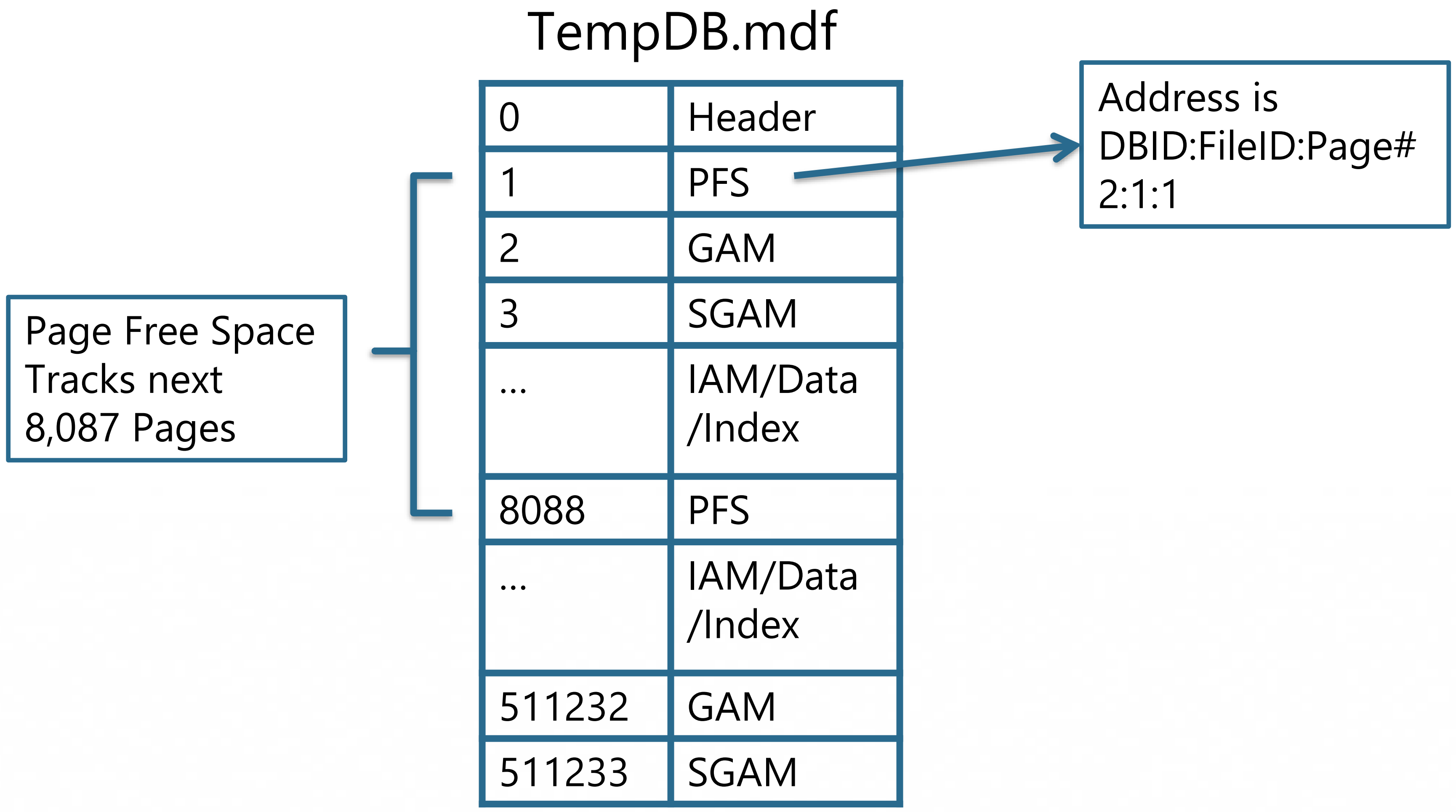
- The wait type is CMEMTHREAD or SOS_CACHESTORE spinlock waits.
- This could indicate other issues besides TempDB Temp Table caching.

The Roles of Allocation Pages

PFS and IAM pages are used to determine if a new page or extent is needed

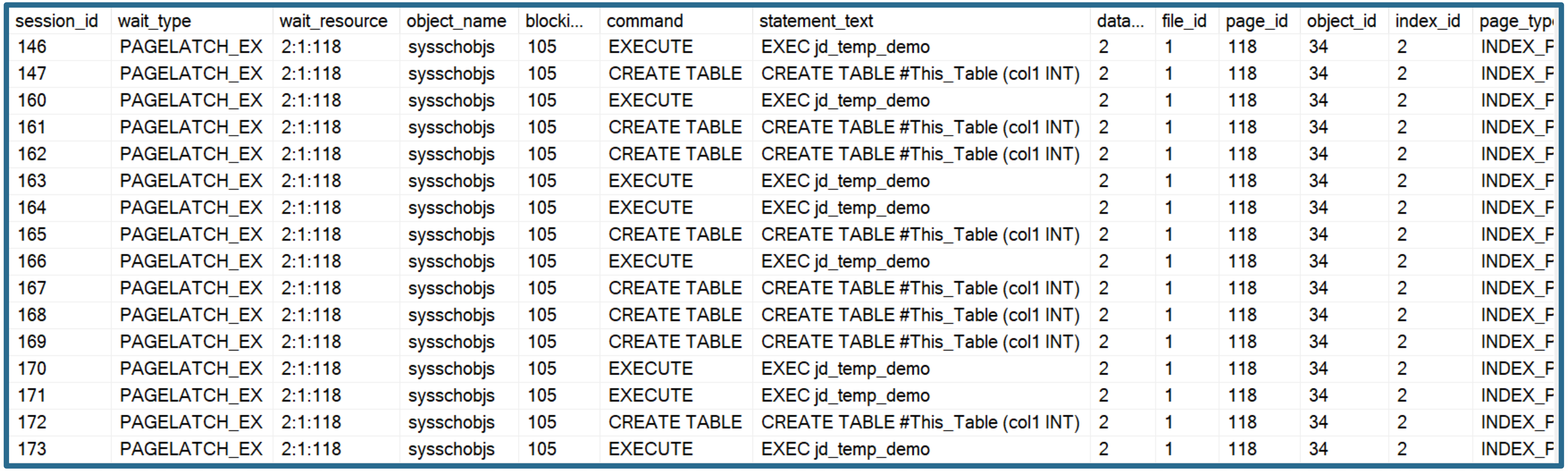
GAM and SGAM pages are used to allocate extents

Database Page Layout



Metadata Contention

The wait type is PAGELATCH_EX on **sysschobjs** in **TempDB..**



session_id	wait_type	wait_resource	object_name	blocki...	command	statement_text	data...	file_id	page_id	object_id	index_id	page_typ
146	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
147	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
160	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
161	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
162	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
163	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
164	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
165	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
166	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
167	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
168	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
169	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
170	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
171	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
172	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
173	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F

Optimizing the TempDB Database

TempDB File Placement

File	Logical name	Physical name	Initial size	File growth
Primary data	tempdev	tempdb.mdf	8 MB	Autogrow by 64 MB until the disk is full
Secondary data files*	temp#	tempdb_mssql_#.ndf	8 MB	Autogrow by 64 MB until the disk is full
Log	templog	templog.ldf	8 MB	Autogrow by 64 megabytes to a maximum of 2 terabytes

If the number of logical processors is less than or equal to eight (8), use the same number of data files as logical processors.

If the number of logical processors is greater than eight (8), use eight data files.

If contention continues, increase the number of data files by multiples of four (4) up to the number of logical processors

Alternatively, make changes to the workload or code.

Optimizing TempDB performance

Consider instant file initialization

Pre-allocate space for all TempDB files

Divide TempDB into multiple data files of equal size

Put the TempDB database on a fast I/O subsystem

Use disk striping if there are many directly attached disks.

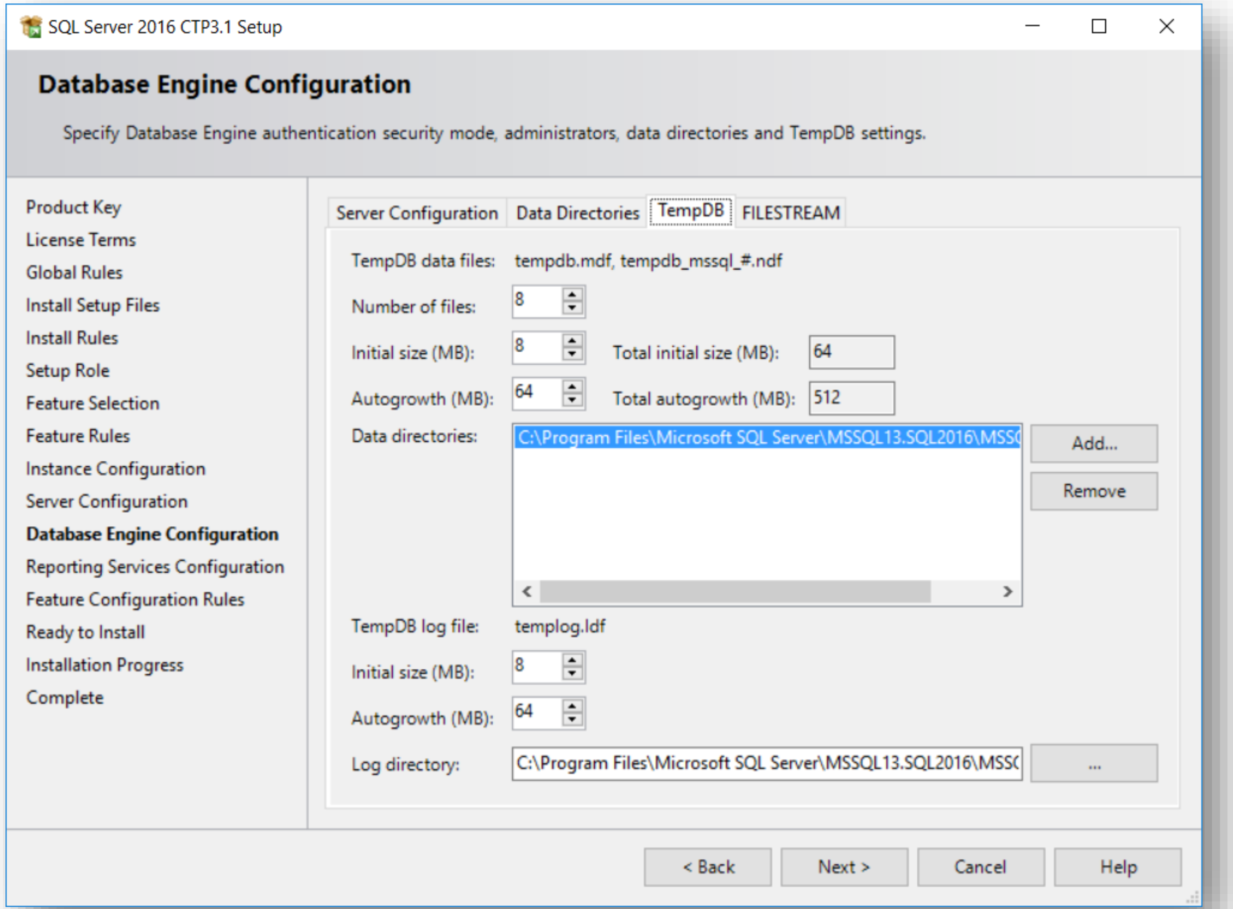
Put the TempDB database on separate disks from user databases

TempDB Database Performance Improvements

Performance improvements in TempDB (2016)

Setup adds multiple TempDB data files during instance installation.

By default, setup adds as many TempDB data files as the logical processor count or eight, whichever is lower.



Database Engine Configuration

Specify Database Engine authentication security mode, administrators, data directories and TempDB settings.

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.SQL2016\MSS

TempDB log file: templog.ldf

Initial size (MB): 8

Autogrowth (MB): 64

Log directory: C:\Program Files\Microsoft SQL Server\MSSQL13.SQL2016\MSS

Performance improvements in TempDB (2016)

Trace Flags behavior enabled by default for TempDB

- 1117 (Grow all files in a filegroup evenly)
- 1118 (Avoid mixed extents and only use full extents)

Temporary tables and table variables are cached.

Improved allocation page latching.

- PAGELATCH_SH on metadata allocation instead of PAGELATCH_EX

Logging overhead for TempDB is reduced.

Performance improvements in TempDB (2019)

Default	Opt-in
• Temp table cache improvements • Concurrent PFS updates	• Memory-Optimized TempDB Metadata

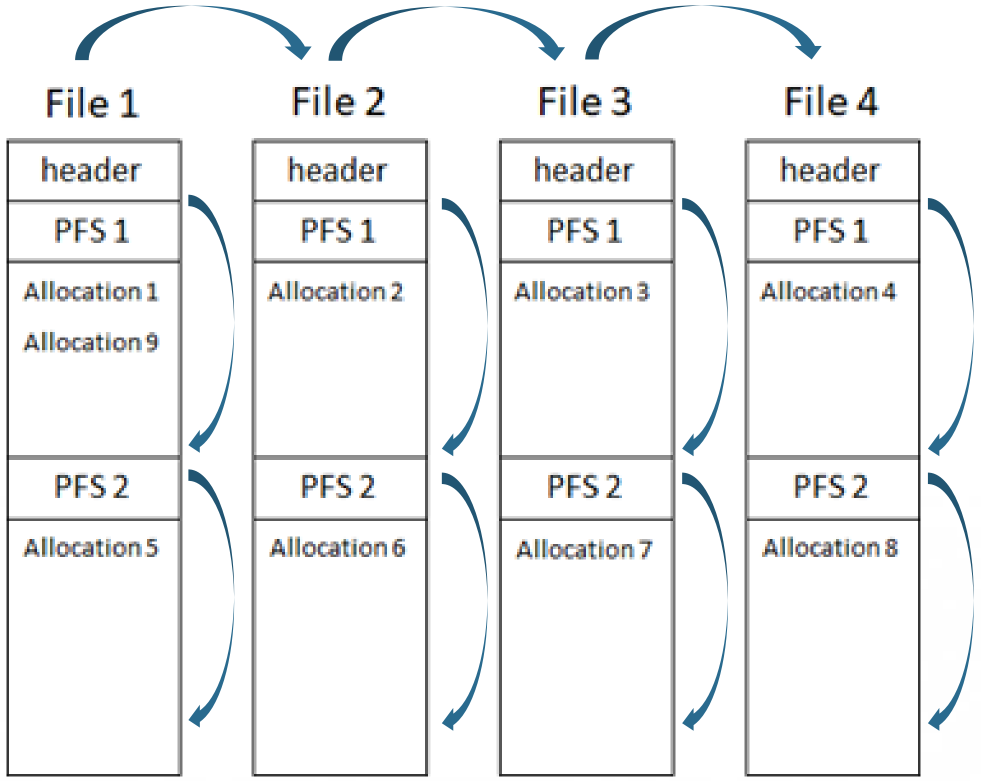
TEMPDB Files, Trace Flags, and Updates by Pam Lahoud

How (and When) To: Memory Optimized TempDB

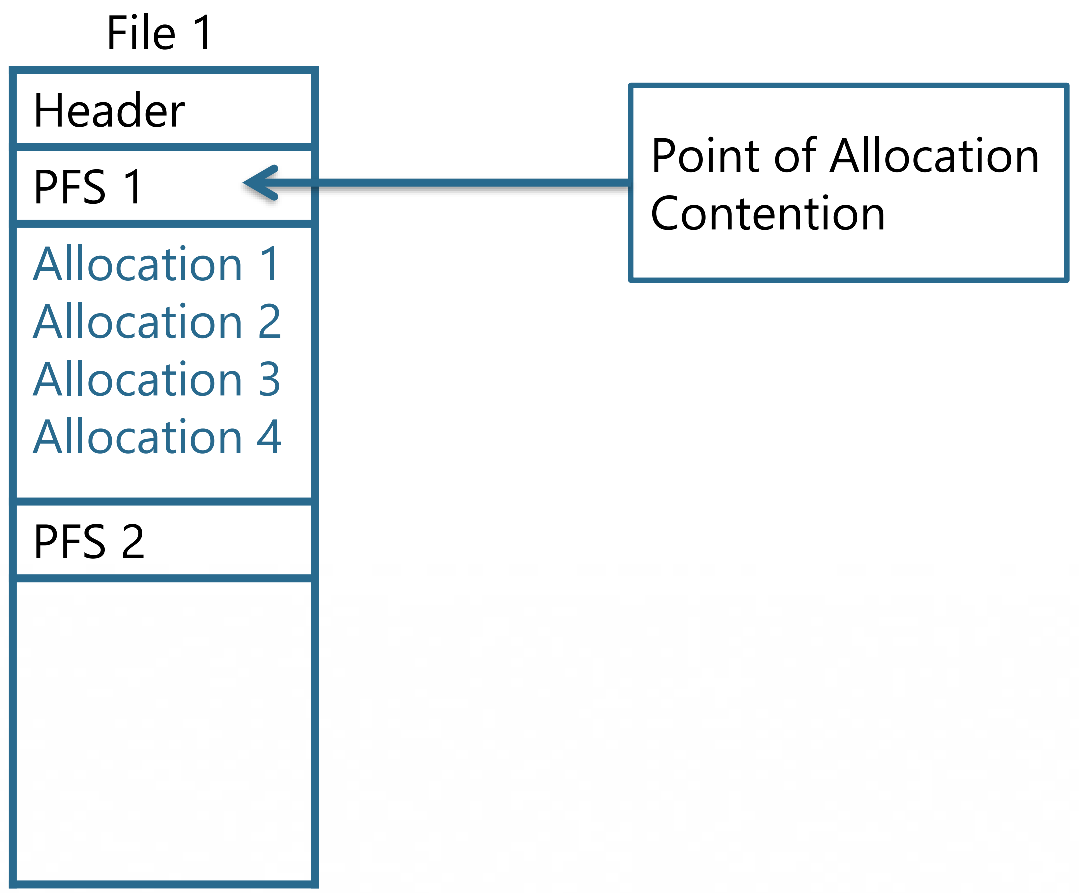
Concurrent PFS Pages in TempDB (2019)

New algorithm uses round robin between files, and between PFS pages within the files.

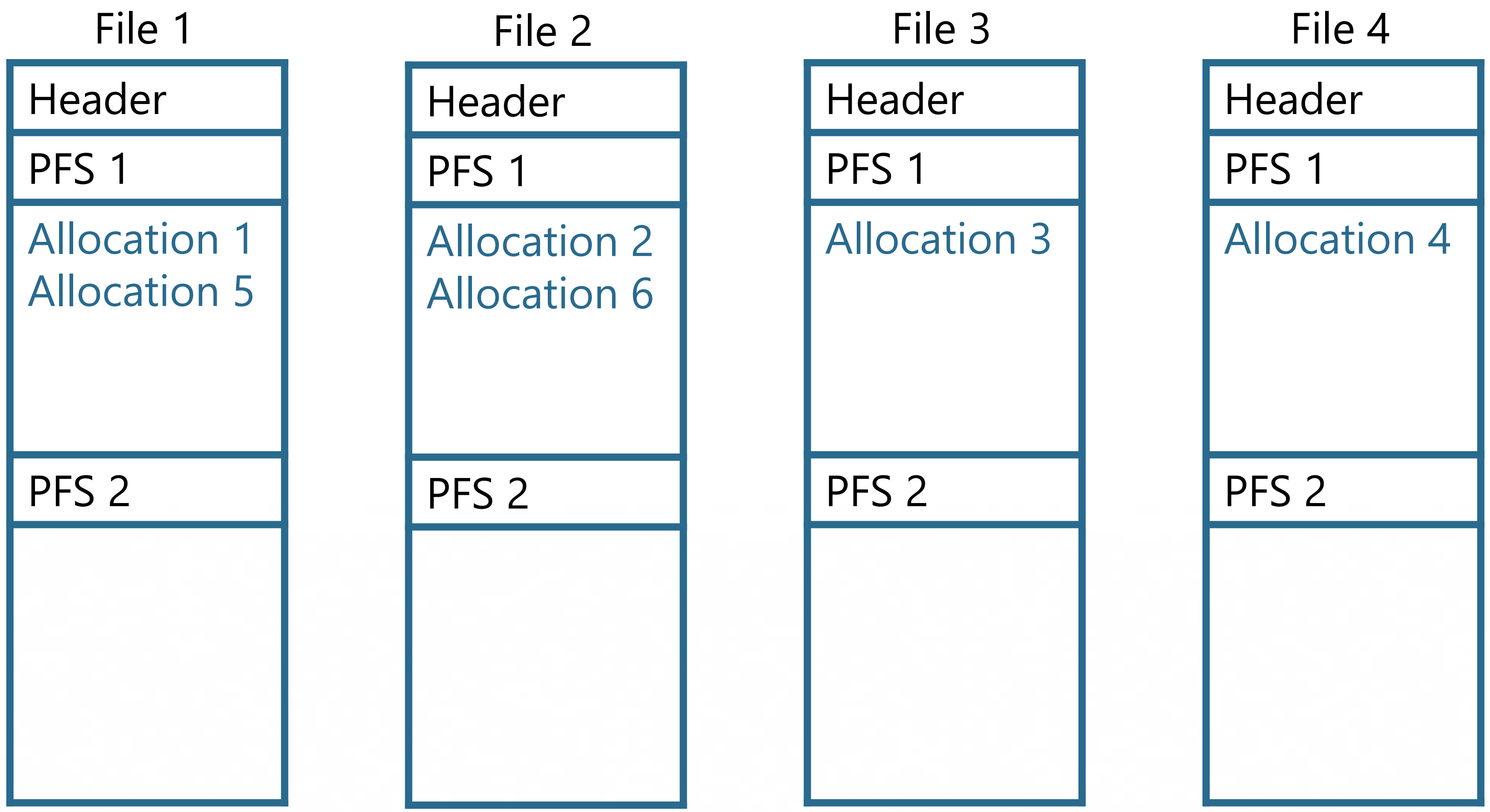
With this change, not only will increasing the number of files help with PFS contention but also increasing the size of the files.



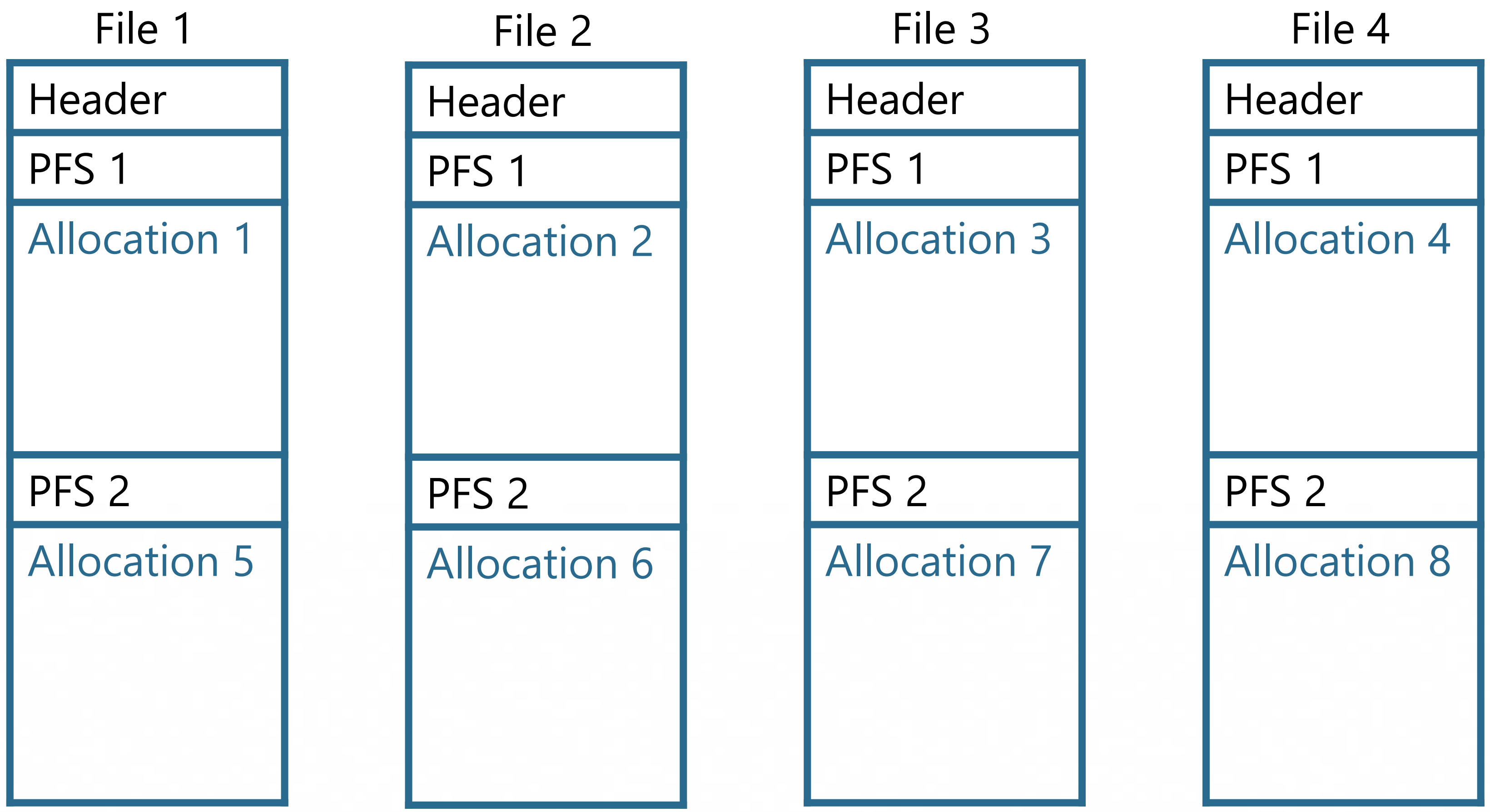
PFS Page Allocation (Single File: Pre-2019)



PFS Page Round Robin (Multiple Files: Pre-2019)

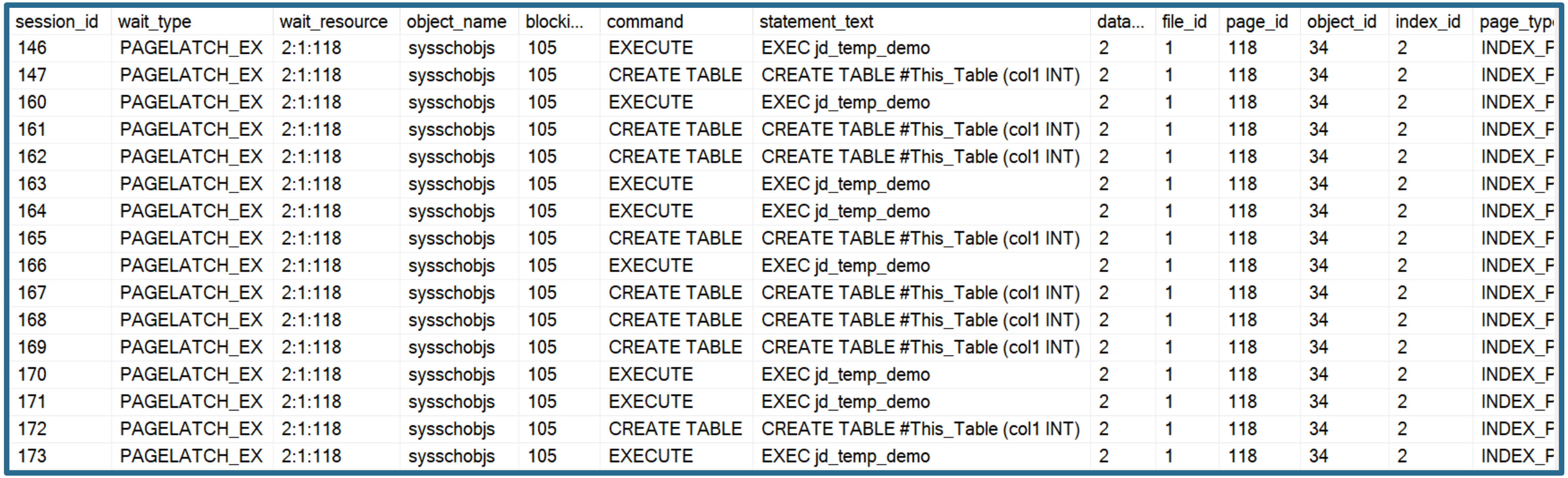


PFS Page Round Robin (Multiple Files: 2019)



Metadata Contention

The wait type is PAGELATCH_EX on **sysschobjs** in **TempDB**..



session_id	wait_type	wait_resource	object_name	blocki...	command	statement_text	data...	file_id	page_id	object_id	index_id	page_typ
146	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
147	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
160	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
161	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
162	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
163	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
164	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
165	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
166	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
167	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
168	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
169	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
170	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
171	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F
172	PAGELATCH_EX	2:1:118	sysschobjs	105	CREATE TABLE	CREATE TABLE #This_Table (col1 INT)	2	1	118	34	2	INDEX_F
173	PAGELATCH_EX	2:1:118	sysschobjs	105	EXECUTE	EXEC jd_temp_demo	2	1	118	34	2	INDEX_F

Identifying Metadata Contention

```sql
SELECT er.session_id, er.wait_type, er.wait_resource,
OBJECT_NAME(page_info.[object_id],page_info.[database_id]) as [object_name],
er.blocking_session_id, er.command,
SUBSTRING(st.text, (er.statement_start_offset/2) + 1,
    ((CASE statement_end_offset
        WHEN -1 THEN DATALENGTH(ST.text)
        ELSE er.statement_end_offset END
            - er.statement_start_offset)/2) + 1) AS statement_text,
page_info.database_id,
page_info.[file_id],
page_info.page_id,
page_info.[object_id],
page_info.index_id,
page_info.page_type_desc
FROM sys.dm_exec_requests AS er
  CROSS APPLY sys.dm_exec_sql_text(er.sql_handle) as st
  CROSS APPLY sys.fn_PageResCracker(er.page_resource) AS r
  CROSS APPLY sys.dm_db_page_info(r.db_id, r.file_id, r.page_id,'DETAILED') AS
       page_info;
```

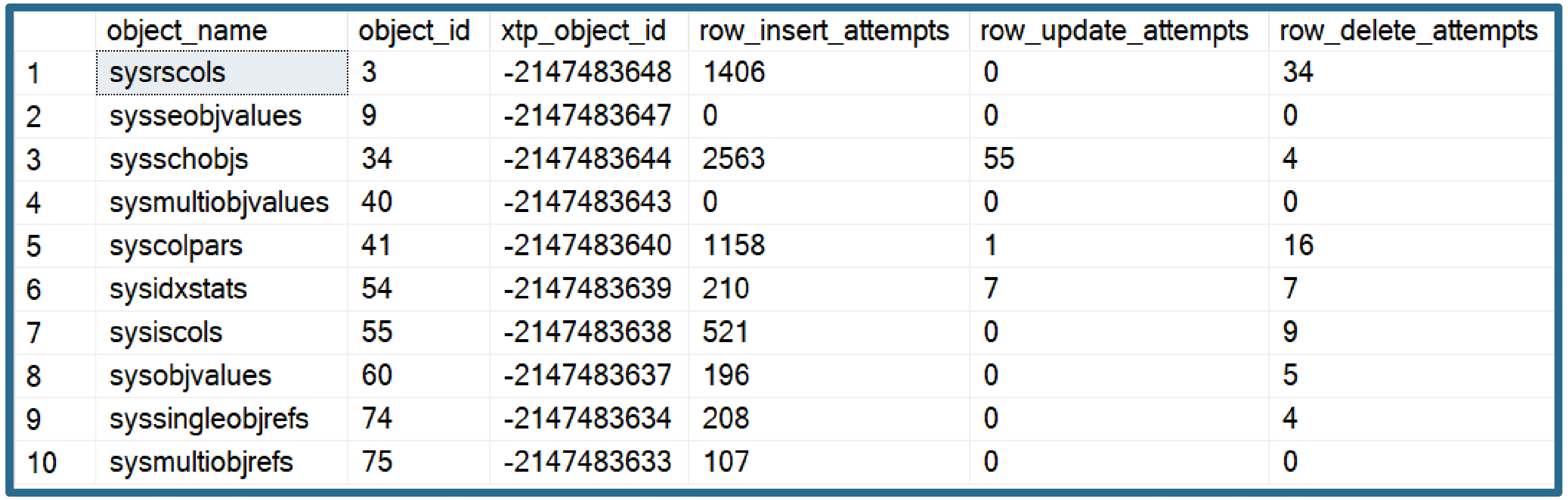
TempDB Memory Optimized Metadata Tables

Enable Memory Optimized TempDB Metadata Tables

```sql
ALTER SERVER CONFIGURATION SET MEMORY_OPTIMIZED TEMPDB_METADATA = ON;
```

Search for Memory Optimized Tables

```sql
SELECT OBJECT_NAME(object_id) as [object_name], * FROM sys.dm_db_xtp_object_stats;
```



	object_name	object_id	xtp_object_id	row_insert_attempts	row_update_attempts	row_delete_attempts
1	sysrscols	3	-2147483648	1406	0	34
2	sysseobjvalues	9	-2147483647	0	0	0
3	sysschobjs	34	-2147483644	2563	55	4
4	sysmultiobjvalues	40	-2147483643	0	0	0
5	syscolpars	41	-2147483640	1158	1	16
6	sysidxstats	54	-2147483639	210	7	7
7	sysiscols	55	-2147483638	521	0	9
8	sysobjvalues	60	-2147483637	196	0	5
9	syssingleobjrefs	74	-2147483634	208	0	4
10	sysmultiobjrefs	75	-2147483633	107	0	0

Dankie Faleminderit **Shukran** Chnorakaloutioun Hvala Blagodaria

Děkuji **Tak** Dank u Tänan Kiitos **Merci** Danke Ευχαριστώ A dank

Mahalo תודה. **Dhanyavād** Köszönöm Takk Terima kasih **Grazie** Grazzi

Thank you!

감사합니다 Paldies Choukrane Ačiū **Благодарам** ありがとうございました

谢谢 Баярлалаа **Dziękuję** Obrigado Mulţumesc **Спасибо** Ngiyabonga

Ďakujem Tack Nandri Kop khun **Teşekkür ederim** Дякую Хвала Diolch