TEMPDB CONTENTION
HOW TO IDENTIFY AND RESOLVE IT

Haripriya Naidu
SQL DBA
She/Her

About me

- SQL Database Administrator – 10 years
- Focus on
  - SQL Server Engine Internals
  - Performance Tuning
  - Server/DB Operations
- Actively participate in
  - NESQL
  - SQL Saturday Boston
  - Multiple online learning communities
- AWS Certified Solutions Architect
- Rookie of the Year 2023 @Work

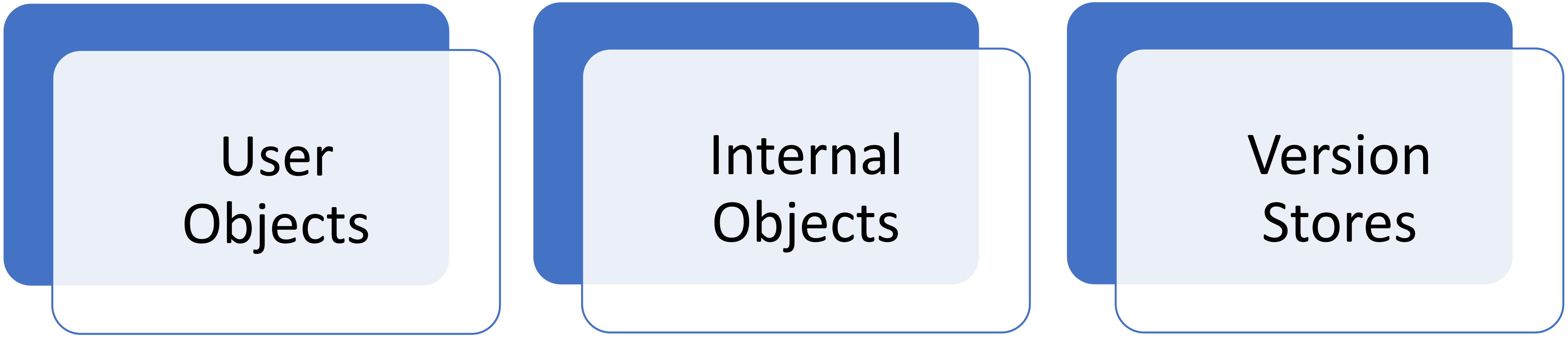
Agenda

- Uses of TempDB
- Types of Contention
- Identifying Contention
- Ways to Resolve

Why learn about TempDB Contention

- Troubleshoot Slowness
- Concurrency Management

What runs on TempDB



User Objects

Internal Objects

Version Stores

User objects

- Local and Global Temp Tables - #table, ##table
- Local and Global Temp Stored Procedures - #proc, ##proc
- Table Variables - @table
- Appear in sys.all_objects view

Internal objects

- Worktables to store intermediate results for
  - Sorts, Hash Joins, Aggregates
  - Spools, Cursors
  - INSTEAD OF triggers
  - Store Service Broker messages in transit
  - Online Index Rebuild when SORT_IN_TEMPDB is ON
- Users cannot create these internal objects

Version Stores

- Used for Row versioning for the following:
  - Snapshot isolation levels
  - Online index operations
  - AFTER triggers
  - MARS(Multiple active result sets)

Types of Contention

- Object Allocation Contention – DML
- Metadata Contention – DDL
- Temp Table Cache Contention

Object Allocation Contention - DML

- In any database, when there are insert, update, delete operations on user objects, they need a latch (lightweight lock) on PFS, GAM, SGAM pages on a datafile.
- In tempdb, multiple insert, update, delete sessions run concurrently on temporary objects, waiting for a latch on PFS, GAM, SGAM pages on a datafile, causing contention

Datafile – tempdb.mdf

Page 0: File Header

Page 1: Page Free Space (PFS)

Page 2: Global Allocation Map (GAM)

Page 3: Shared Global Allocation Map (SGAM)

Page 4: Unused

Page 5: Unused

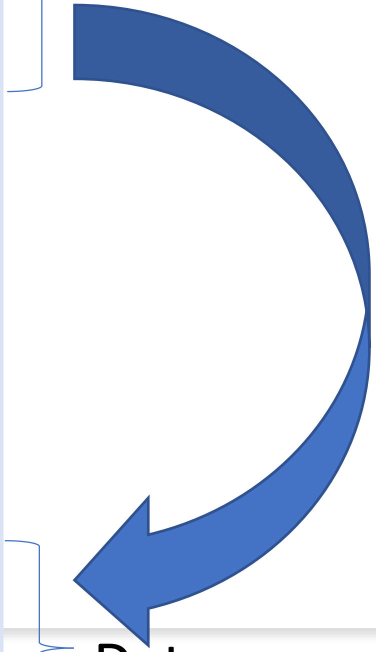
Page 6: Differential Change Map (DCM)

Page 7: Bulk Change Map (BCM)

Data pages

… <more data pages>

Tracking pages



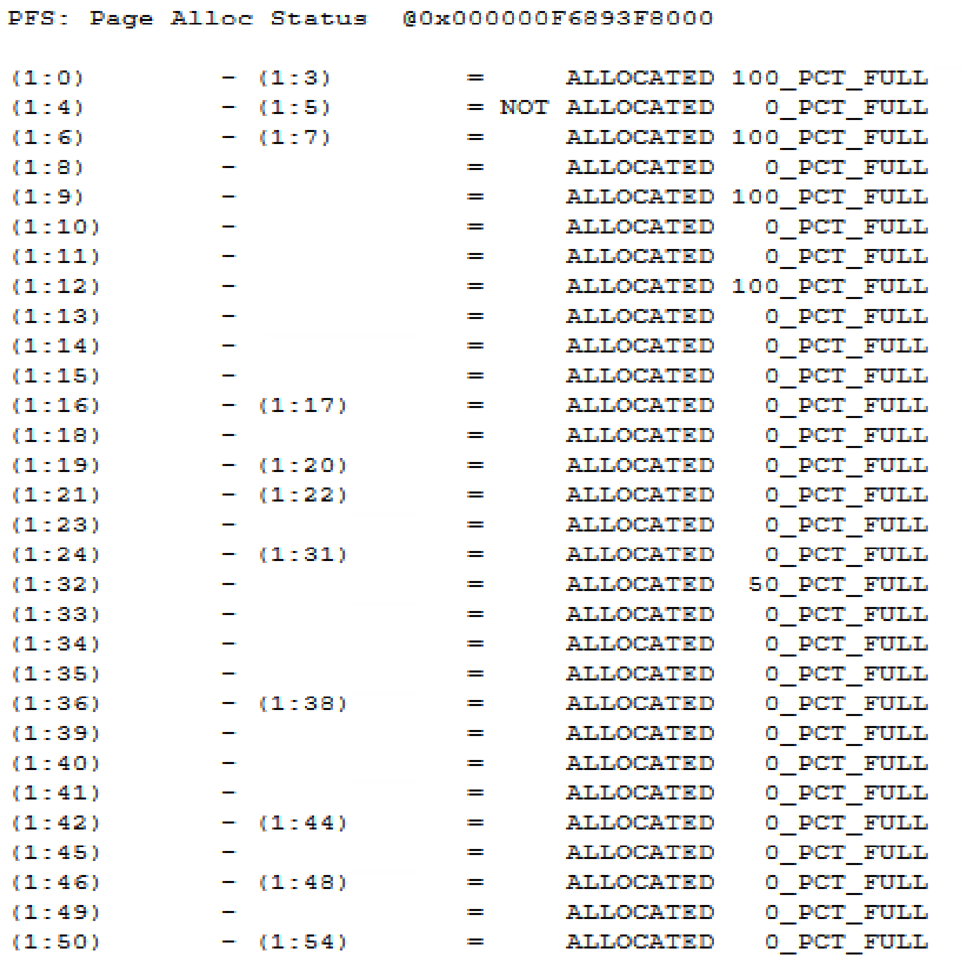
- Track the status of data pages
- Used when SQL Server needs to allocate space

Data pages

Page Free Space (PFS)

- Tracks free space available for each page
- All operations need a latch on PFS page to know which page has space

Page 1: PFS



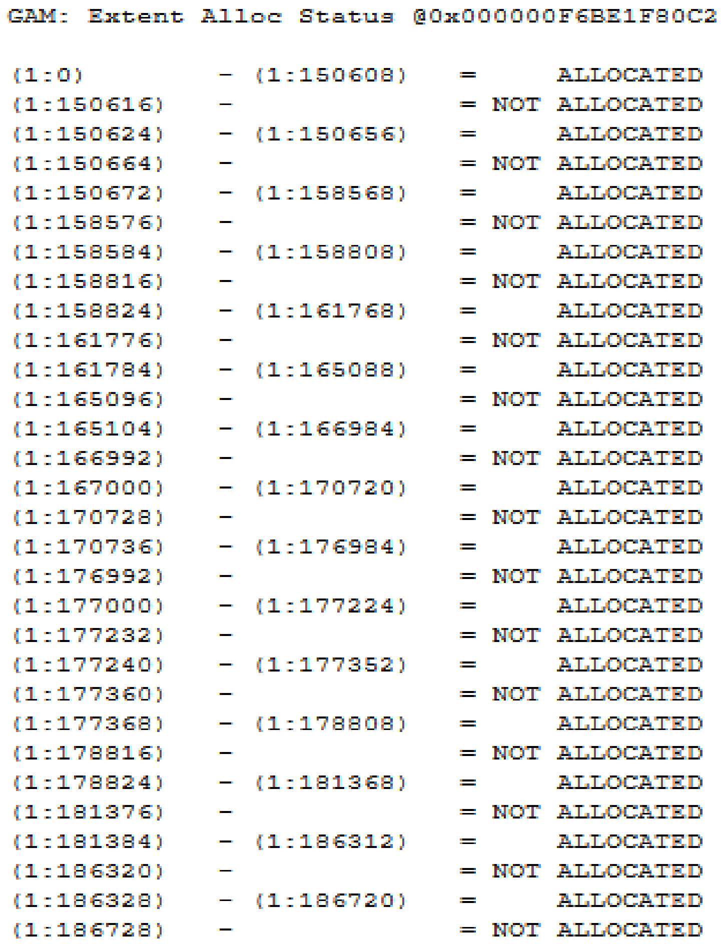
```
PFS: Page Alloc Status  @0x000000F6893F8000

(1:0)        - (1:3)        =     ALLOCATED 100_PCT_FULL
(1:4)        - (1:5)        = NOT ALLOCATED   0_PCT_FULL
(1:6)        - (1:7)        =     ALLOCATED 100_PCT_FULL
(1:8)        -              =     ALLOCATED   0_PCT_FULL
(1:9)        -              =     ALLOCATED 100_PCT_FULL
(1:10)       -              =     ALLOCATED   0_PCT_FULL
(1:11)       -              =     ALLOCATED   0_PCT_FULL
(1:12)       -              =     ALLOCATED 100_PCT_FULL
(1:13)       -              =     ALLOCATED   0_PCT_FULL
(1:14)       -              =     ALLOCATED   0_PCT_FULL
(1:15)       -              =     ALLOCATED   0_PCT_FULL
(1:16)       - (1:17)       =     ALLOCATED   0_PCT_FULL
(1:18)       -              =     ALLOCATED   0_PCT_FULL
(1:19)       - (1:20)       =     ALLOCATED   0_PCT_FULL
(1:21)       - (1:22)       =     ALLOCATED   0_PCT_FULL
(1:23)       -              =     ALLOCATED   0_PCT_FULL
(1:24)       - (1:31)       =     ALLOCATED   0_PCT_FULL
(1:32)       -              =     ALLOCATED  50_PCT_FULL
(1:33)       -              =     ALLOCATED   0_PCT_FULL
(1:34)       -              =     ALLOCATED   0_PCT_FULL
(1:35)       -              =     ALLOCATED   0_PCT_FULL
(1:36)       - (1:38)       =     ALLOCATED   0_PCT_FULL
(1:39)       -              =     ALLOCATED   0_PCT_FULL
(1:40)       -              =     ALLOCATED   0_PCT_FULL
(1:41)       -              =     ALLOCATED   0_PCT_FULL
(1:42)       - (1:44)       =     ALLOCATED   0_PCT_FULL
(1:45)       -              =     ALLOCATED   0_PCT_FULL
(1:46)       - (1:48)       =     ALLOCATED   0_PCT_FULL
(1:49)       -              =     ALLOCATED   0_PCT_FULL
(1:50)       - (1:54)       =     ALLOCATED   0_PCT_FULL
```

GAM (Global Allocation Map)

Page 2: GAM

- Tracks if an uniform extent is allocated or not
- 1 extent = 8 pages
- Uniform extent = all 8 pages belong to same object
- All operations need a latch on GAM page to know which page is allocated or not



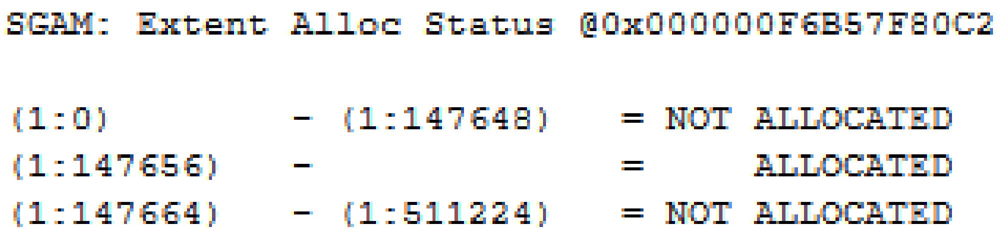
```
GAM: Extent Alloc Status @0x000000F6BE1F80C2

(1:0)        - (1:150608)  =     ALLOCATED
(1:150616)   -             = NOT ALLOCATED
(1:150624)   - (1:150656)  =     ALLOCATED
(1:150664)   -             = NOT ALLOCATED
(1:150672)   - (1:158568)  =     ALLOCATED
(1:158576)   -             = NOT ALLOCATED
(1:158584)   - (1:158808)  =     ALLOCATED
(1:158816)   -             = NOT ALLOCATED
(1:158824)   - (1:161768)  =     ALLOCATED
(1:161776)   -             = NOT ALLOCATED
(1:161784)   - (1:165088)  =     ALLOCATED
(1:165096)   -             = NOT ALLOCATED
(1:165104)   - (1:166984)  =     ALLOCATED
(1:166992)   -             = NOT ALLOCATED
(1:167000)   - (1:170720)  =     ALLOCATED
(1:170728)   -             = NOT ALLOCATED
(1:170736)   - (1:176984)  =     ALLOCATED
(1:176992)   -             = NOT ALLOCATED
(1:177000)   - (1:177224)  =     ALLOCATED
(1:177232)   -             = NOT ALLOCATED
(1:177240)   - (1:177352)  =     ALLOCATED
(1:177360)   -             = NOT ALLOCATED
(1:177368)   - (1:178808)  =     ALLOCATED
(1:178816)   -             = NOT ALLOCATED
(1:178824)   - (1:181368)  =     ALLOCATED
(1:181376)   -             = NOT ALLOCATED
(1:181384)   - (1:186312)  =     ALLOCATED
(1:186320)   -             = NOT ALLOCATED
(1:186328)   - (1:186720)  =     ALLOCATED
(1:186728)   -             = NOT ALLOCATED
```

SGAM (Shared Global Allocation Map)

- Tracks if a mixed extent is allocated or not
- Mixed extent = 8 pages belong to different objects
- All operations need a latch on SGAM page to know which page is allocated or not

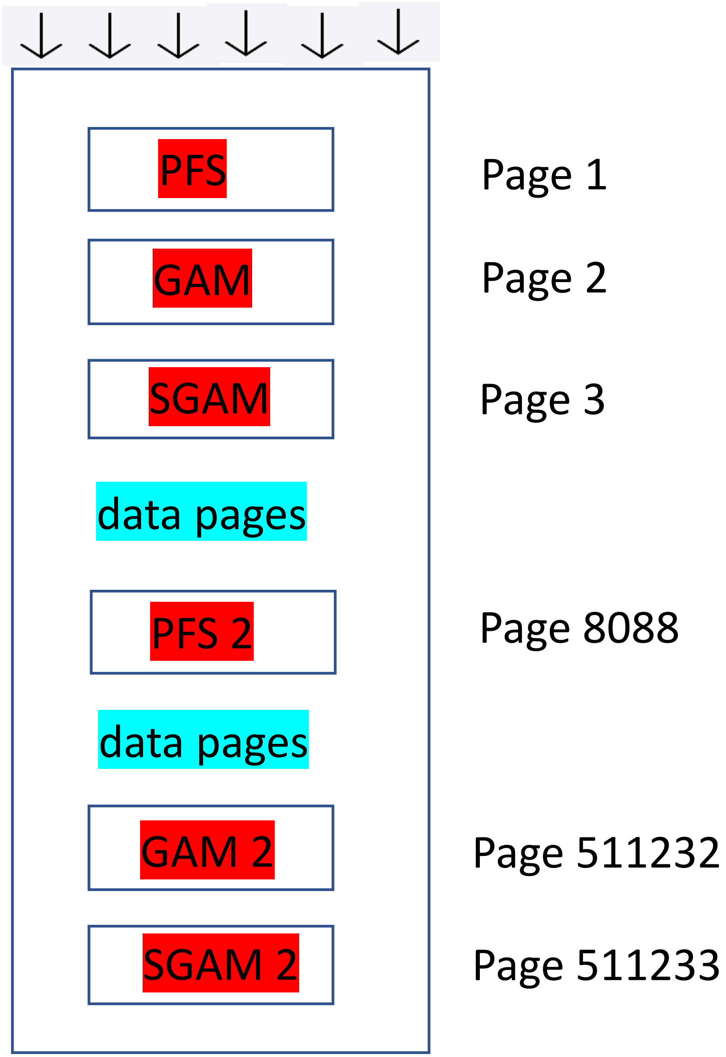
Page 3: SGAM



```
SGAM: Extent Alloc Status @0x000000F6B57F80C2

(1:0)          - (1:147648)    = NOT ALLOCATED
(1:147656)     -               =     ALLOCATED
(1:147664)     - (1:511224)    = NOT ALLOCATED
```

Multiple transactions hit on a datafile



PFS Page 1

GAM Page 2

SGAM Page 3

data pages

PFS 2 Page 8088

data pages

GAM 2 Page 511232

SGAM 2 Page 511233

Tempdb.mdf

https://gohigh.substack.com/p/pfs-gam-sgam-pages

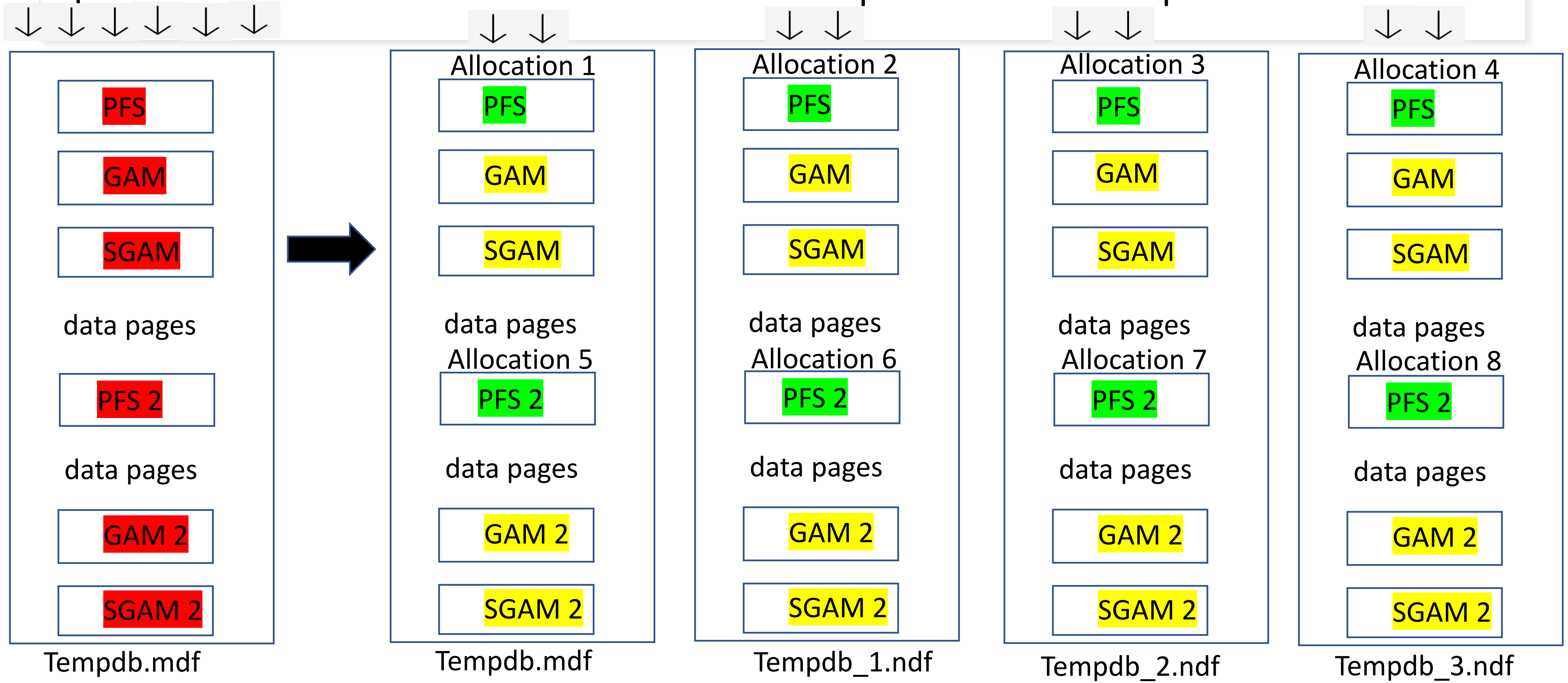
Identify Object Allocation Contention

- PAGELATCH_UP, PAGELATCH_EX waits on PFS and (S)GAM pages
- How to find them:
  - 2:FileID:1 or 2:FileID:<any PFS or (S)GAM page>
- Perfmon counters:
  - Access Methods::Worktables created/sec
  - Access Methods::Workfiles created/sec
  - Number > 200

Resolve Object allocation contention

Multiple Transactions

Transactions are spread across multiple datafiles



PFS
GAM
SGAM
data pages
PFS 2
data pages
GAM 2
SGAM 2
Tempdb.mdf

Allocation 1
PFS
GAM
SGAM
data pages
Allocation 5
PFS 2
data pages
GAM 2
SGAM 2
Tempdb.mdf

Allocation 2
PFS
GAM
SGAM
data pages
Allocation 6
PFS 2
data pages
GAM 2
SGAM 2
Tempdb_1.ndf

Allocation 3
PFS
GAM
SGAM
data pages
Allocation 7
PFS 2
data pages
GAM 2
SGAM 2
Tempdb_2.ndf

Allocation 4
PFS
GAM
SGAM
data pages
Allocation 8
PFS 2
data pages
GAM 2
SGAM 2
Tempdb_3.ndf

Resolve Object allocation contention

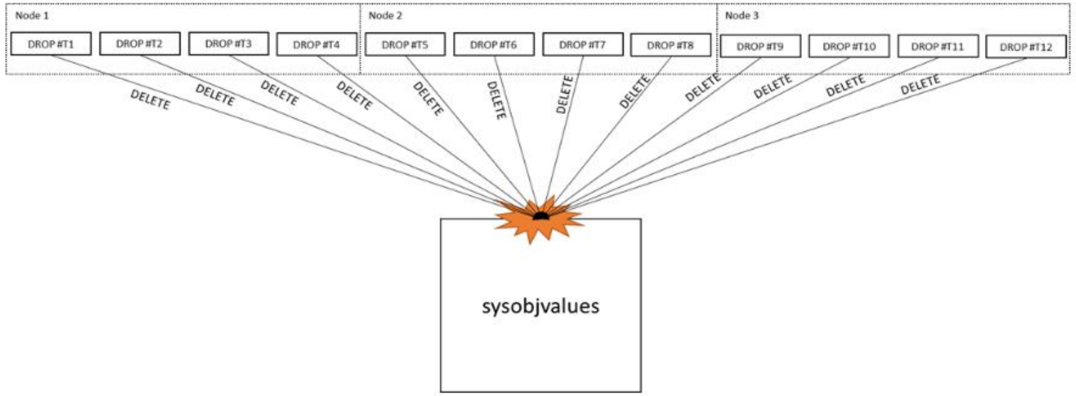
- Round Robin
  - between datafiles
  - between PFS pages starting in SQL Server 2017 CU7
- Create 1 data file per logical processor up to 8 logical processors
- Additional files in numbers of 4
- Create with same size and same autogrowth
- Proportionally fills datafiles based on empty space

Resolve Object allocation contention – by SQL Server version

- Prior to SQL 2016: Turn ON Trace flags 1117, 1118
- SQL 2016 and onwards: Trace Flags are ON by default
- SQL 2019: PFS page is updated with a shared latch instead of exclusive latch
- SQL 2022: GAM and SGAM pages are updated with a shared latch
- Follow best practices across all versions

Metadata contention – DDL

- In any database, when there are create, alter, drop operations on user objects, they need a latch on system object pages on a datafile.
- In tempdb, multiple create, alter, drop sessions run concurrently on temporary objects, waiting for a latch on system object pages, causing contention



https://learn.microsoft.com/en-us/archive/blogs/sql_server_team/tempdb-files-and-trace-flags-and-updates-oh-my

Identify Metadata Contention

- PAGELATCH_EX, PAGELATCH_UP waits on system catalog tables - such as sysobjvalues and sysschobjs
- How to find them 2:FileID:<system page>
- Perfmon counters:
  - SQLStatistics::PageLatch Wait
  - Temp Tables Creation Rate
  - Temp Tables Destruction Rate

Resolve Metadata contention

- Don’t drop/truncate temp tables
- Cache temporary objects:
  - Don’t alter temp table DDL statements after creation
  - Create temp objects within SP, trigger, function
- Enable Memory-Optimized TempDB Metadata starting SQL 2019
  - CU2 moves sysallocunits

Temp table cache contention

- Contention on cache for Temp table memory objects
- Why?

  Inefficient usage of Hash tables in cache

Identify/Resolve Temp table cache contention

- CMEMTHREAD waits
- SOS_CACHESTORE spinlock waits
- Open a Microsoft case, needs advanced debugging

DEMO

Summary: Object Allocation Contention

- Observe: PFS, GAM, SGAM pages
- Wait Type: PAGELATCH_UP, PAGELATCH_SH
- Resolution:
    - Multiple Datafiles
    - Upgrade to SQL Server Version 2022

Summary: Metadata Contention

- Observe: System Object pages
- Wait Type: PAGELATCH_EX, PAGELATCH_SH
- Resolution:
  - Don’t drop Temp objects
  - CACHE Temp objects
  - Turn ON in-memory TempDB metadata feature in SQL Server Version 2019 and above

Summary: Temp Table Cache Contention

- Observe: Huge memory objects in cache
- Wait Type: CMEMTHREAD, SOS_CACHESTORE
- Resolution:

Why learn about TempDB Contention

Troubleshoot Slowness

Concurrency Management

Resources

- TempDB: The Good, The Bad, and The Ugly by Pam Lahoud
  - https://youtu.be/5jpfy19yl7k?si=vmgf3zqxokurpeo0
- OStress utility: https://learn.microsoft.com/en-us/troubleshoot/sql/tools/replay-markup-language-utility
- Klaus Aschenbrenner: https://www.sqlservercentral.com/blogs/improved-temp-table-caching-in-sql-server-2014
- Pro SQL Server Internals book by Dmitri Korotkevitch

Reach Out For More Questions

- Email: haripriya.naidu1@gmail.com
- GitHub: https://github.com/haripriyasb/tempdb
- Substack: https://gohigh.substack.com
- LinkedIn: https://www.linkedin.com/in/haripriya-naidu1215

Thank You!