1999997 - FAQ: SAP HANA Memory

Version	862	**Type**	SAP Knowledge Base Article
Language	English	**Master Language**	English
Release Status	Released to Customer	**Category**	How To
Component	HAN-DB (SAP HANA Database)	**Released On**	24.04.2020

Please find the original document at **https://launchpad.support.sap.com/#/notes/1999997**

Symptom

You have questions related to the SAP HANA memory.

You experience a high memory utilization or out of memory dumps.

Environment

SAP HANA

Cause

Resolution

1. Which indications exist for SAP HANA memory problems?

Tracefiles with the following naming convention are created:

<service>_<host>.<port>.rtedump.<timestamp>.oom.trc
<service>_<host>.<port>.rtedump.<timestamp>.oom_memory_release.trc
<service>_<host>.<port>.rtedump.<timestamp>.compositelimit_oom.trc
<service>_<host>.<port>.rtedump.<timestamp>.after_oom_cleanup.trc
<service>_<host>.<port>.emergencydump.<timestamp>.trc (if memory related errors like "allocation failed" are responsible)

The following error messages can indicate OOM situations. Be aware that some of the errors can also be issued in other scenarios. To make sure that they are really memory related, you have to check the related trace file.

```
-9300: no more memory -10760: memory allocation failed -10108: Session has been reconnected
2: general error: allocation failed
4: cannot allocate enough memory 12. Cannot allocate memory 129: transaction rolled back by an
internal error: Memory allocation failed 129: transaction rolled back by an internal error:
exception during deltalog replay. 129: transaction rolled back by an internal error: TableUpdate
failed 129: transaction rolled back by an internal error: exception 1000002: Allocation failed ;
$size$=1191936; $name$=TableUpdate; $type$=pool; $inuse_count$=2180; $allocated_size$=8180736;
$alignment$=16# 129: transaction rolled back by an internal error: TrexUpdate failed on table
<table_name> with error: commitOptimizeAttributes() failed with rc=6900, Attribute engine
failed;object=<object_name>$delta_1$en, rc=6900 - enforce TX rollback 129: transaction rolled
back by an internal error: TrexUpdate failed on table '<table_name>' with error: Attribute load
failed;index=<table_name>en,attribute='$trexexternalkey$' (207), rc=6923 - enforce TX rollback
129: transaction rolled back by an internal error: TrexUpdate failed on table '<table_name>'
with error: AttributeEngine: not enough memory, rc=6952 - enforce TX rollback 403: internal
error 1024: Allocation failed $REASON$ 2048: column store error: search table error: [2] message
not found 2048: column store error: search table error: [9] Memory allocation failed 2048:
column store error: search table error: [1999] general error (no further information available)
2048: column store error: search table error: [2575] flatten scenario failed; Allocation failed
2048: column store error: search table error: [6900] Attribute engine failed 2048: column store
error: search table error: [6923] Attribute load failed 2048: column store error: search table
error: [6952] Error during optimizer search 2048: column store error: search table error: [6952]
AttributeEngine: not enough memory 2048: column store error: [2450] error during merge of delta
index occurred 2048: column store error: [6924] Attribute save failed 2048: column store error:
merge delta index error: [6924] Attribute save failed 3584: distributed SQL error: [9] Memory
```

```
allocation failed 3584: distributed SQL error: [2617] executor: plan operation execution failed
with an exception 3587: invalid protocol or service shutdown during distributed query execution:
[2613] executor: communication problem plan <plan> failed with rc 9: Error executing physical
plan: Memory allocation failed persistence error: exception 70029020: ltt::exception caught
while operating on DISK_NCLOB:<id>:<id> exception 1000002: Allocation failed ; $size$=<size>;
$name$=Page; $type$=pool; $inuse_count$=<count>; $allocated_size$=<size> Error 423 has occurred
on the current database connection "DEFAULT". Database error text: AFL error: OmsTerminate:
error=-28530, liveCache BAD_ALLOCATION in <routine> liveCache ERROR -1028000 exception 71000004
```

Delta merges / table optimizations (SAP Note 2057046) fail with the following errors:

```
2048 column store error: [2009] Memory allocation failed 2048 column store error: [2201] Not
enough persistent memory available
2048 column store error: [2450] Error during merge of delta index occurred 2048 column store
error: [2484] not enough memory for table optimization 2048 column store error: [6923] Attribute
load failed 2048 column store error: [6924] Attribute save failed 2048 column store error:
[6952] AttributeEngine: not enough memory
```

Backups fail with errors like:

```
Allocation failed ; $size$=<size>; $name$=ChannelUtils::copy; $type$=pool; $inuse_count$=0;
$allocated_size$=0
```

The following entries in the SAP HANA database trace files (SAP Notes 2380176, 2467292) exist:

```
mergeDeltaIndex failed for <schema>:<table> (<id>) rc=245 memAllocSystemPages failed with rc=12,
12 (Cannot allocate memory)
```

A consistency check with CHECK_TABLE_CONSISTENCY (SAP Note 2116157) fails with:

```
5088: The check for some table failed due to memory allocation failure
(ERROR_MEMORY_ALLOCATION_FAILED)
```

The following thread states and locks indicate issues in the memory area (SAP Note 1999998):

Thread state	Lock name
Mutex Wait	HugeAlignmentPool
Mutex Wait	LimitOOMReport
Mutex Wait	PoolAllocator-MemoryPool
Semaphore Wait	IpmmTaskWait
Semaphore Wait	MemoryReclaim

The following SAP HANA alerts indicate problems in the memory area:

Alert	Name	Description
1	Host physical memory usage	Determines what percentage of total physical memory available on the host is used. All processes consuming memory are considered, including non-SAP HANA processes.
6	Address space usage	Determines the address space consumption
12	Memory usage of name server	Determines what percentage of allocated shared memory is being used by the name server on a host.
40	Total memory usage of column store tables	Determines what percentage of the effective allocation limit is being consumed by individual column-store tables as a whole (that is, the cumulative size of all of a table's columns and internal structures).

43	Memory usage of services	Determines what percentage of its effective allocation limit a service is using.
44	Licensed memory usage	Determines what percentage of licensed memory is used.
45	Memory usage of main storage of column store tables	Determines what percentage of the effective allocation limit is being consumed by the main storage of individual column-store tables.
55	Columnstore unloads	Determines how many columns in columnstore tables have been unloaded from memory. This can indicate performance issues.
64	Total memory usage of table-based audit log	Determines what percentage of the effective allocation limit is being consumed by the database table used for table-based audit logging.
68	Total memory usage of row store	Determines the current memory size of a row store used by a service.
81	Cached view size	Determines how much memory is occupied by cached view
116	Transparent Huge Pages status	Determines if Transparent Huge Pages (THP) are activated which can cause issues for the HANA Database.
602	Streaming project physical memory usage	Determines what percentage of total physical memory available on the host is used for the streaming project.
701	Agent memory usage	Determines what percentage of total available memory on agent is used.

SQL: "HANA_Configuration_MiniChecks" (SAP Notes 1969700, 1999993) returns a potentially critical issue (C = 'X') for one of the following individual checks:

Check ID	Details
M0230	Current memory utilization (%)
M0231	Time since memory utilization > 95 % (h)
M0240	Current swap utilization (GB)
M0241	Time since swap utilization > 1 GB (h)
M0242	Swap out (MB, last day)
M0245	Swap space size (GB)
M0264	Virtual memory map count limit
M0410	Current allocation limit used (%)
M0411	Current allocation limit used by tables (%)
M0413	Time since allocation limit used > 80 % (h)
M0415	Curr. max. service allocation limit used (%)
M0417	Time since service alloc. limit used > 80 % (h)
M0420	Heap areas currently larger than 50 GB
M0421	Heap areas larger than 100 GB (last day)
M0422	Heap areas larger than 200 GB (history)
M0423	Heap areas with potential memory leak
M0425	Pool/RowEngine/CpbTree leak size (GB)
M0426	Row store table leak size (GB)
M0430	Number of column store unloads (last day)

M0431	Time since last column store unload (days)
M0435	Number of shrink column unloads (last day)
M0437	Size of unloaded columns (GB, last day)
M0438	Memory reclaim activity (s / day)
M0439	Memory reclaim maximum duration (s)
M0440	Shared memory utilization of nameserver (%)
M0445	Number of OOM events (last hour)
M0450	Tables with memory LOBs > 2 GB
M0453	Size of non-unique concat attributes (GB)
M0454	Size of non-unique concat attributes (%)
M0455	Unused large non-unique concat attributes
M0456	Unused large non-unique row store indexes
M0460	Calc engine cache utilization (%)
M0462	Caches with large size
M0463	Planning engine runtime objects mem. share (%)
M0470	Heap allocators with many instantiations
M0472	Booked vs. allocated memory (%)
M0480	Address space utilization (%)
M0530	Shared memory row store size (GB)
M0611	Memory profiler started
M0645	Number of OOM tracefiles (last day)
M0746	Histories with primary key
M0747	Number of zero entries in HOST_SQL_PLAN_CACHE
M0748	History of M_CS_UNLOADS collected

SQL: "HANA_TraceFiles_MiniChecks" (SAP Note 2380176) reports one of the following check IDs:

Check ID	Details
T0300	Memory allocation failed
T0302	Out of memory (OOM)
T0304	Out of memory (OOM) exception
T0306	Operating system cannot allocate memory
T0308	Statement memory limit reached
T0310	Resource container shrink
T0312	Leaking allocator destroyed
T0319	Shared memory: No space left on device
T0320	Dubious NUMA configuration

SQL: "HANA_Tables_ColumnStore_UnloadsAndLoads" (UNLOAD_REASON = 'LOW MEMORY') (SAP Note 1969700) shows significant amounts of column unloads for the considered time frame.

2. How can I collect information about the current SAP HANA memory consumption?

SAP Note 1969700 provides the following SQL statements to collect information related to the current SAP HANA memory allocation:

SQL statement name	Description
SQL: "HANA_Memory_Caches_Overview"	Overview of existing SAP HANA caches (SAP Note 2502256)
SQL: "HANA_Memory_Components"	High level overview of current and historic memory consumption
SQL: "HANA_Memory_ContextMemory"	Current context memory utilization, useful to map used memory to connections
SQL: "HANA_Memory_MemoryObjects"	Current memory objects in SAP HANA resource container (heap + row store); only used for areas taking advantage of caching, so temporary SQL areas like Pool/itab aren't part of it.
SQL: "HANA_Memory_Overview"	Provides information about current memory allocation (including heap, row store, column store, allocation limit and license limit)
SQL: "HANA_Memory_TopConsumers"	Lists the top memory consumers (e.g. tables and heap areas)

SAP Note 1698281 provides a Python script that can be used to collect detailed SAP HANA memory requirements. In order to get precise data, columns are actually loaded into memory rather than only relying on estimations.

If you don't have SQL access (e.g. on the secondary site of a SAP HANA system replication environment), you can use the operating system tool hdbcons (SAP Note 2222218) and 'mm l -S' to display the allocators sorted by the inclusive memory size. Sorting by the more important exclusive size in use is not possible. Starting with SAP HANA 1.0 SPS 11 you can also query this information via _SYS_SR_SITE_<site_name>. See SAP Note 1999880 ("Is it possible to monitor remote system replication sites on the primary system?") for details.

3. How can I collect information about the historic SAP HANA memory consumption?

SAP Note 1969700 provides the following SQL statements to collect information related to the historic SAP HANA memory allocation:

SQL statement name	Description
SQL: "HANA_Memory_Reclaims"	Information about historic reclaim operations (i.e. defragmentations or shrinks)
SQL: "HANA_Memory_TopConsumers" SQL: "HANA_Memory_TopConsumers_TimeSlices"	List historic top memory consumers (e.g. tables and heap areas)
SQL: "HANA_Memory_OutOfMemoryEvents"	Overview of out-of-memory (OOM) situations since last startup
SQL: "HANA_SQL_SQLCache"	Starting with SAP HANA Rev. 102.01 memory information is available in the SQL cache (if memory tracking is activated) that can be evaluated.
SQL: "HANA_SQL_ExpensiveStatements"	Lists memory consumption of executed SQL statements (SPS 08) Relevant output columns: MEM_USED_GB, MEM_PER_EXEC_GB Both expensive SQL statement

	trace and statement memory tracking needs to be activated, see "Is it possible to limit the memory that can be allocated by a single SQL statement?" in this SAP Note for more information.

4. Which important memory areas exist?

The following memory areas are most important:

Memory Area	Context	Level	Details
Physical memory	operating system	global	Total amount of memory physically available on host level (typically RAM)
Virtual memory	operating system	process	Total amount of memory allocated by all processes held both in physical memory and in paging area on disk
Resident memory	operating system	process	Total amount of memory allocated by all processes held in physical memory, large allocations are usually fine (SAP Note 2081473)
Allocated memory	SAP HANA	process	Total amount of memory allocated by the SAP HANA processes, limited by the configurable SAP HANA global allocation limit Less relevant for SAP HANA memory analysis because allocated, but unused memory can be re-used when required
Used memory	SAP HANA	process	Total amount of memory in use by the SAP HANA processes, relevant to understand SAP HANA memory footprint
Shared memory	SAP HANA	global	Memory that can be accessed by different processes, e.g.: Specific row store components (tables, catalog, free) Nameserver topology
Heap memory	SAP HANA	process	Memory exclusively accessible by threads of a single process (e.g. indexserver), e.g.: Column store Row store indexes Intermediate results Temporary structures SAP HANA page cache
Code	SAP HANA	global	Code
Stack	SAP HANA	process	Stack

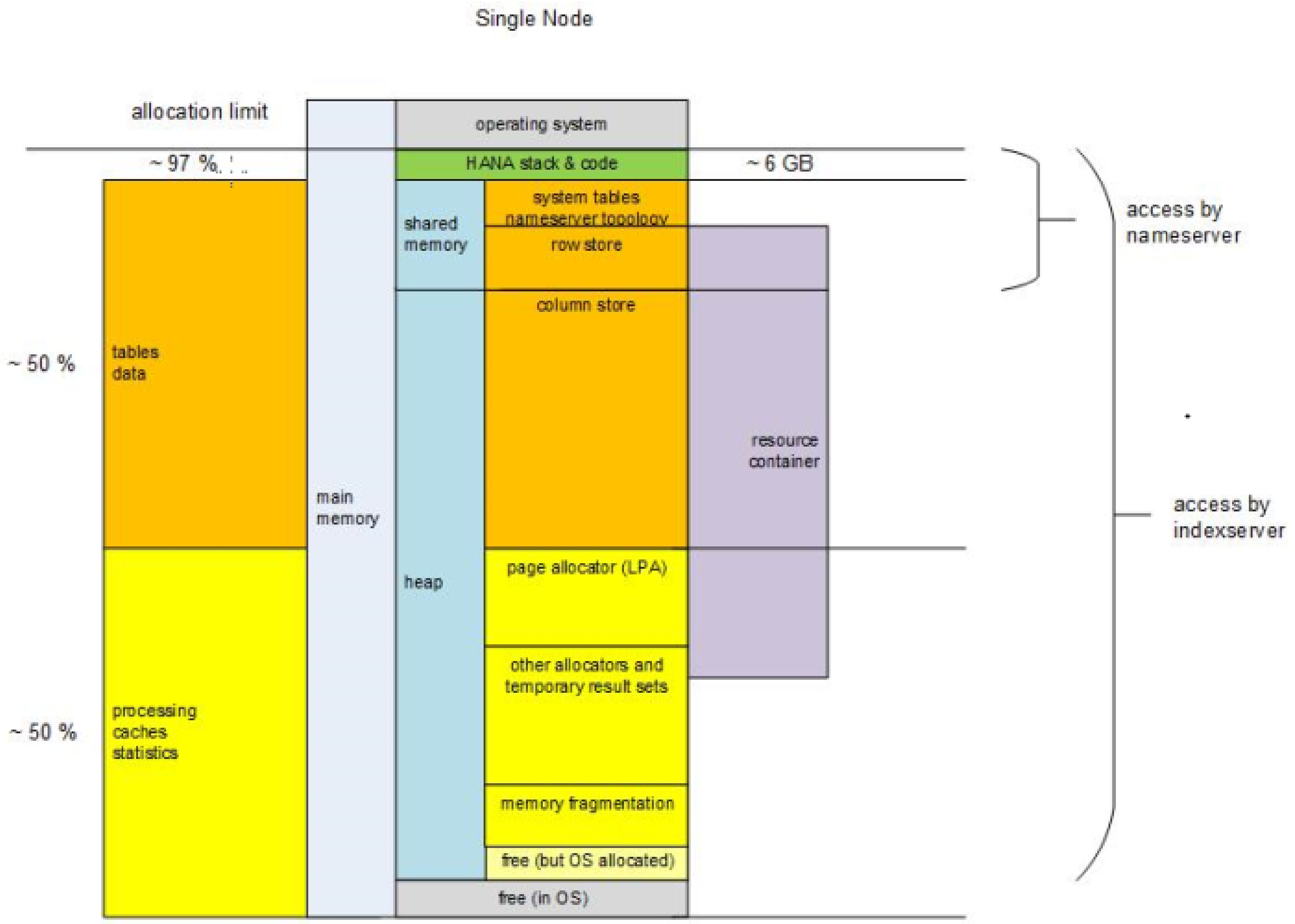
In normal SAP HANA environments no paging happens and SAP HANA is the only major memory allocator on the host. The following conditions are typically met:

- Physical memory > virtual memory
- Virtual memory = resident memory >= allocated memory
- Allocated memory = shared memory + allocated heap memory
- Used memory = shared memory + used heap memory
- Code, stack: Usually negligible sizes

For efficiency reasons SAP HANA frees allocated memory in a "lazy" way and so the allocated memory can grow up to the available memory and the global allocation limit while the used memory remains at a much lower level.

From a memory analysis perspective we can usually focus on the used memory and assume that the allocated memory is released whenever required.

The following picture illustrates the general SAP HANA memory structure:



5. What does SAP HANA do if memory becomes scarce?

Unlike other databases (e.g. Oracle: PGA in memory -> PSAPTEMP on disk) SAP HANA doesn't allocate disk space if certain operations require more memory than available. Instead the following actions are taken:

Action	hdbcons	Details
Reclaim - return free memory to OS		Free memory segments are returned to operating system. This is helpful in cases where a SAP HANA memory request is larger than the individually available segments in the SAP HANA heap memory. The operating system is able to perform a defragmentation and provide larger segments afterwards. This operation isn't recorded in database trace (SAP Note 2380176). You can determine reclaim activities using the "mm ipmm -d" option of hdbcons (SAP Note 2222218). The following information indicates reclaim at the specified point in time: T=2016-12-08 04:42:43.722 ...: Process <pid> requested self compaction A self compaction is a reclaim that is triggered by the process itself.
Reclaim - defragmentation	mm gc -f	Garbage collection is triggered so that allocated memory is defragmented and freed for re-use. It is executed automatically when not sufficient free memory is available. For a manual / explicit control of memory garbage collection see question "What is memory memory garbage collection?" below. See SAP Note SAP Note 2169283 for more information related to SAP HANA garbage collection. Releasing the memory back to the operating system requires the IPMM lock, so memory allocations can be blocked (e.g. with "ReclaimMemory" locks, see SAP Note 1999998). This operation isn't recorded in database trace (SAP Note 2380176). You can determine reclaim activities using the "mm ipmm -d" option of hdbcons (SAP Note 2222218). The following information indicates

		reclaims at the specified point in time: T=2016-12-08 04:42:43.722 ...: Process <pid> requested self compaction
Reclaim - resource container shrink	resmans	Resource container is shrunk: Non-critical heap areas are reduced (e.g. the SAP HANA page cache Pool/PersistenceManager/PersistentSpace(0)/DefaultLPA/Page or compiled L code) Column store unloads are triggered (SAP Note 2127458); this activity can significantly impact the performance. Automatic shrinks are recorded in the database trace (SAP Note 2380176) and can be found by checking for the following information: Information about shrink
Termination of transactions		Transactions are terminated with error if their memory requests can no longer be fulfilled.
OOM dump		An out-of-memory (OOM) dump is written (if the time defined in parameter global.ini -> [memorymanager] -> oom_dump_time_delta is exceeded since the last OOM dump)

Memory garbage collection and shrinks are locally done by the thread that detects the need for these tasks. In the following cases a specific MemoryCompcator thread is used for that purpose (SAP Note 2114710):

- Memory garbage collection and shrink requests coming from another SAP HANA service
- Memory garbage collection triggered by explicitly configured parameters global.ini -> [memorymanager] -> gc_unused_memory_threshold_abs and global.ini -> [memorymanager] -> gc_unused_memory_threshold_rel.

SQL: "HANA_Memory_Reclaims" (SAP Note 1969700) can be used to display reclaim runtimes and processed memory sizes.

6. Which parameters can be used to limit the SAP HANA memory consumption?

The following parameters can be used to limit the overall or process-specific SAP HANA memory allocation:

Parameter	Unit	Details
global.ini -> [memorymanager] -> global_allocation_limit	MB	This parameter limits the overall DRAM memory consumption of SAP HANA. It covers DRAM memory allocated in context of the fast restart option, but it doesn't cover persistent memory (SAP Note 2700084). The default value depends on the available physical memory and the SAP HANA revision level: SAP HANA 1.0 SPS 06 and below: 90 % of physical memory SAP HANA 1.0 SPS 07 and higher: 90 % of first 64 GB, 97 % of remaining physical memory
global.ini -> [memorymanager] -> persistent_memory_global_allocation_limit	MB	This parameter limits the memory consumption in persistent memory. See SAP Note 2700084 for more information.
<service>.ini -> [memorymanager] -> allocationlimit	MB %	This parameter limits the memory consumption of the related SAP HANA process (<service>). If "%" is specified at the end of the parameter value (without preceeding blank), the value is interpreted as percentage of RAM, otherwise it is interpreted as MB. The standalone statistics server uses a value of "5%" per default. All other services including indexserver use the following allocation limit per default: Rev. <= 1.00.92: 90 % of physical memory Rev. >= 1.00.93: global_allocation_limit As an example, SAP Note 1862506 suggests an increase of the allocation limit of

		the standalone statistics server to "10%", "15%" or "20%" in order to come around OOM situations caused by the default 5 % limit. This setting can also be used to limit the memory usage of a tenant database in MDC environments (SAP Note 2175606).

Normally there is no need to touch these settings and there are other solutions to come around memory issues.

7. How can I analyze problems related to the SAP HANA memory consumption?

SAP Note 1840954 describes steps to analyze and resolve SAP HANA memory issues.

SAP Note 1984422 describes how to analyze an out of memory (OOM) dump file.

SAP Note 2222718 provides a decision-tree approach for analyzing problems in the SAP HANA memory area.

The SAP HANA Troubleshooting and Peformance Analysis Guide at SAP HANA Troubleshooting and Performance Analysis Guide covers - among others - the analysis of memory related issues.

8. Is it possible to extend the physical memory of a SAP HANA machine?

In general the configured physical memory depends on factors like hardware, scenario and available CPUs and must not be changed. SAP Note 1903576 describes when and how you can apply for an exception.

9. Which options exist to reduce the risk of SAP HANA memory issues?

The following options exist to reduce the risk of SAP HANA memory issues:

Action / Feature	Details
Cleanup of technical tables	Make sure that house-keeping is set up for technical, administration and communication tables so that they don't consume unnecessary memory. See SAP Note 2388483 for more information.
Archiving	Implement archiving strategies for business data. Have a look at the Information Lifecycle Management area for more details.
S/4HANA	S/4HANA significantly reduces redundancy of table data (e.g. FI: elimination of BSEG index tables like BSIS, BSID, BSAS and BSAD) and so it has a positive impact on the memory footprint. See the Simplification List for S/4HANA for further details.
Hybrid LOBs	Hybrid LOBs are not loaded into memory when the size exceeds a defined limit, so it is usually beneficial for memory consumption if you take advantage of this feature. SAP Note 1994962 describes how columns defined as memory LOBs can be converted to hybrid LOBs. SAP ABAP table columns with LRAW data type are mapped to either LOB or VARBINARY. As VARBINARY always has to be loaded into memory, this can have an effect on the memory utilization. See SAP Note 2220627 ("Is VARBINARY also a LOB Type?") for more information. SAP Note 2375917 describes how a VARBINARY column can be converted into a LOB in order to save memory.
Reduction of number of indexes	Check for indexes with high memory requirements (e.g. using SQL: "HANA_Indexes_Overview", ORDER_BY = 'SIZE' from SAP Note 1969700) and check if you can drop some of these indexes. A focus can be put in the following areas: Secondary indexes that were created in order to optimize the performance of non-HANA databases. BW: If DSOs are changed from "standard" to "write-optimized", a primary index is no longer

	required. BW: Check if you can flag the property “Allow duplicate records” of write-optimized DSOs because this will eliminate the need for multicolumn key indexes (/BIC/A…00KE). Check if large fulltext indexes (SAP Note 2800008) are really required. For example, a large index REPOSRC~SRC (on a column with name $_SYS_SHADOW_DATA) may exist to support the ABAP Sourcecode Search (SAP Note 1918229) and can be removed via transaction SFW5. Dropping indexes can significantly impact performance, so you should test the effects carefully before permanently dropping indexes.
Transition from multi-column to single-column indexes	Multi-column indexes require much more memory than single-column indexes, because an additional internal column (concat attribute) needs to be created. Check for indexes with high memory requirements (e.g. using SQL: "HANA_Indexes_Overview", ORDER_BY = 'SIZE' from SAP Note 1969700) and check if you can redefine some multi-column indexes to single-column indexes. Often it is a good compromise to define an index only on the most selective column. Further columns like MANDT would significantly increase the memory requirements.
Reduction of concat attributes	Concat attributes are specific internal columns that can be created for various reasons. Some of them may no longer be required. See SAP Note 1986747 for more information. You can run SQL: "HANA_Indexes_ColumnStore_RedundantConcatAttribute s" (SAP Note 1969700) in order to define redundant concat attributes - i.e. multiple concat attributes created on the identical set of columns (with the identical order) and use the generated DROP_COMMAND to drop one of these duplicates. A typical reason for this behavior for SID tables in BW environments is described in SAP Note 2376550
Paged attributes	Paged attributes are columns that can be loaded into the memory piece-wise. All columns apart from primary key and internal columns can be defined as paged attributes. For more details see SAP Note 1871386.
Inverted hash indexes	As of SAP HANA 1.0 SPS 09 you can reduce the size of multi-column indexes using the inverted hash feature. This can reduce the size of the internal concat attribute that is required for multi-column indexes. See SAP Note 2109355 for more information.
Inverted individual indexes	Starting with SAP HANA 2.0 SPS 03 primary keys and unique indexes can be defined as inverted individual indexes which eliminate the need to have a potentially large concat attribute and so the index size can be significantly reduced. See SAP Note 2600076 for more details.
Move large tables to column store	Table data is compressed efficiently in column store, so moving tables from row store to column store usually reduced the memory allocation significantly. Furthermore table columns are only loaded into the column store memory if required and not during startup. Therefore you can check if large tables exist in row store that can be moved to column store. Be aware that tables with a significant amount of modifications can suffer from performance regressions if they are moved to column store. In case of SAP standard tables you should usually double-check with SAP if the move to the column store is an option.
Analysis of large heap areas	Some heap areas may be larger than required, e.g. due to bugs or inadequate configuration. See question "What can I do if a certain heap allocator is unusually large?" below for more details.
SQL statement optimization	SQL statements processing large amounts of data or accessing data inefficiently can be responsible for a significant memory growth. See SAP Note 2000002 related to SQL statement optimization. See question "Which general optimizations exist for reducing the SQL statement memory requirements?" below for more information.
Transactional problems	Long running transactions or idle cursors can impact the garbage collection and result in a high amount of versions or histories. See SAP Note 2169283 for more information about symptoms, analysis steps and resolutions in the area of garbage collection.
Fragmentation	Fragmentation effects can result in an unnecessary row store size. See SAP Note 1813245

(row store)	for more information on checking the row store fragmentation and reorganizing the row store. Starting with SAP HANA 2.0 SPS 04 the row store is defragmented online and automatically once a certain fragmentation level is reached (SAP Note 2789255).
Fragmentation (heap memory)	See "Can there be fragmentation in the heap memory?" in order to check if there is an unusual high and recurring fragmentation of the heap memory.
Large delta storage	Many records in the delta storage of tables can increase the size of the column store. See SAP Note 2057046 and make sure that delta merges are running properly.
Delta merge and optimize compression	Delta merges (SAP Note 2057046) and optimize compression runs (SAP Note 2112604) temporary require a much larger memory footprint, typically you have to expect that the double size of the underlying table (partition) is needed. Therefore you have to make sure that the size of the table (partitions) is sufficiently small that doubling it is possible without running into a memory bottleneck. Typically you can achieve this by proper data management (see SAP Note 2388483) and by partitioning particularly large tables (SAP Note 2044468).
Column store compression	See SAP Note 2112604 and make sure that the column store tables are compressed optimally.
Unload configuration	It is possible to influence the unload behavior so that less critical objects are unloaded first ("UNLOAD PRIORITY <level>" setting for tables) . The following parameter controls the minimum size of the SAP HANA resource container that needs to be retained (SAP Note 1993128): indexserver.ini -> [memoryobjects] -> unload_lower_bound If this size has reached the defined limit and more memory outside of the resource container is required (e.g. because of an expensive SQL statement), an out-of-memory situation is issued. It is usually not required to configure this parameter because the statement memory limit has similar effects.
Data aging	Data aging (SAP Note 2416490) allows to load only current data into memory while older data is kept on disk. This feature is only available for a defined set of tables.
Dynamic tiering	Using dynamic tiering you can mark data as hot, warm and cold. Typically only hot data resides in the SAP HANA memory. See SAP Note 2140959 for more information related to dynamic tiering.
Smart data access	Based on smart data access SAP HANA can retrieve data from tables in external databases (e.g. Sybase, Oracle or SAP HANA). This reduced the need to load all accessed data into SAP HANA. See SAP Note 2180119 for more information regarding smart data access.
Extension nodes	Starting with SAP HANA 1.00 SPS 12 and 2.00 SPS 01 it is possible to configure extension nodes for tables containing no hot data. By overloading the extension node it is possible to share a limited amount of memory by a high amount of tables. See SAP Note 2415279 for more information.
Table distribution	If some hosts in a scale-out scenario suffer from a high memory consumption you can re-locate tables or table partitions from hosts with a high memory consumption to hosts with a lower memory consumption. See section "Table Distribution in SAP HANA" of the SAP HANA Administration Guide for more information.
Global allocation limit	The following parameter defines the maximum overall memory size which can be allocated by the SAP HANA instance: global.ini -> [memorymanager] -> global_allocation_limit The default value depends on the available physical memory and the SAP HANA revision level: SPS 06 and below: 90 % of physical memory SPS 07 and higher: 90 % of first 64 GB, 97 % of remaining physical memory Particularly on SPS 06 and below and hosts with a lot of memory this can result in a significant amount of unused memory (e.g. SPS 06, 1 TB memory, 90 % allocation limit, up to 900 GB allocated by SAP HANA, 10 GB allocated by

	OS and other components -> 90 GB unused). If you observe a significant amount of permanently unused memory you can increase the global_allocation_limit parameter (e.g. to "95%" or "97%" for SPS 06 and below). Make sure that you don't increase the allocation limit to a value that results in paging. If multiple SAP HANA instances run on the same host, you have to make sure that the sum of all configured global allocation limits doesn't exceed the available memory.
OS configuration	Make sure that the operating system configuration is in line with the SAP recommendations. See SAP Note 2000003 ("How can the configuration and performance of the SAP HANA hardware, firmware and operating system be checked?") for more information. It is particularly important that the ulimit package isn't installed in SLES environments, because it may define address space limitations (e.g. SOFTVIRTUALLIMIT < 100 % in /etc/sysconfig/ulimit). The following command should return nothing, otherwise it has to be uninstalled: rpm -qa | grep ulimit Make sure that no address space limitations are defined for the SAP HANA processes. You can use the following commands to determine the process ID of the indexserver via ps (<indexserver_pid>) and subsequently check for the configured address space limitations: ps -ef | grep indexserver egrep 'Soft|space' /proc/<indexserver_pid>/limits The correct output without limitation looks similar like the following example: Limit Soft Limit Hard Limit Units Max address space unlimited unlimited bytes See also SAP Note 1980196 that discusses OOM errors due to an inadequate setting of the Linux parameter /proc/sys/vm/max_map_count. If multiple SAP HANA instances (or other applications with high memory requirements) run on the same node, make sure that the overall assigned memory (e.g. the global allocation limits for the SAP HANA instances) doesn't exceed the available physical memory. See SAP Note 2123782 which suggests a pagepool size reduction from 16 GB to 4 GB in Lenovo / GPFS environments. Make sure that the limit for stack is not set to a high / unlimited value (SAP Note 2488924) as it can result in a significant address space consumption.
Strict NUMA memory binding	If the operating system issues on OOM although there is sufficient memory available, an erroneous strict NUMA memory binding of SAP HANA processes can be responsible. See SAP Note 2358255 for details. This issue is fixed with Rev. 122.02. See SAP Note 2470289 for more information related to NUMA in SAP HANA environments.
SAP HANA patch level	The memory allocation of certain heap areas is SAP HANA patch level dependent. Newer revision levels may include optimizations that reduce the memory allocation. Therefore it is generally useful to make sure that a reasonably new revision level is implemented.
Scale-out layout	Using fewer hosts with a larger amount of physical memory each will reduce the risk that specific SQL statements with a high memory requirement will result in OOM situations, because there is a larger amount of available memory on each host. So for example 2 hosts with 1 TB memory each would have a lower risk of OOM situations compared to 8 hosts with 256 GB each.
Statistics server optimizations	See SAP Note 2147247 (-> "How can the memory requirements of the statistics server be minimized?") for details.
BW DTP delta initialization request optimization	If you face a high memory consumption related to DTP activities in BW, you can check SAP Note 2230080 for possible optimizations.
Bypassing SAP HANA bugs	Make sure that you are on reasonably new SAP HANA Revision levels and avoid situations that can cause memory related issues due to SAP HANA bugs. Particularly consider the following scenarios: Impacted Revisions Details 1.00.90 - 1.00.97.03 1.00.100 - 1.00.102.00 When a column store table (partition) reaches the 2 billion record limit (SAP Note 2154870) a SAP HANA overflow bug can result in extremely high memory allocation requests like:

	Failed to allocate 2305843008945258496 byte. Failed to allocate 18446744073667608592 byte. As a consequence SAP HANA will run into an out-of-memory situation even if significant amounts of memory are still available. Therefore follow the general strong recommendations and take appropriate actions (e.g. data reduction or partitioning) to avoid that a table (partition) reaches the 2 billion record limit. various Check "What can I do if a certain heap allocator is unusually large?" in order to identify SAP HANA bugs that are responsible for memory leaks and other reasons of unnecessary high memory allocation. 1.00.110 - 1.00.112.05 1.00.120 - 1.00.122.01 If the row store size (shared memory) is significantly larger than the total size of row store tables, you should check if the SAP HANA bug described in SAP Note 2362759 applies (memory freed by delete operations is no longer re-used). <= 1.00.122.01 A bug in context of abapSysTimezone calls can result in erroneous very high memory allocations (SAP Note 2740826), e.g.: Failed to allocate 3098286339661321 byte. <= 1.00.122.14 <= 2.00.012.03 2.00.020 A bug in shared memory accounting in MDC environments (SAP Note 2101244) can result in operating system related OOM situations that could have been prevented if SAP HANA had performed reclaims / shrinks. See SAP Note 2588395 for more information. <= 2.00.024.00 If the total memory size of a workload class is limited, unjustified OOMs can happen. See SAP Note 2629536 for more information. See also "Is the SAP HANA memory information always correct?" -> M_CONTEXT_MEMORY below for scenarios where a wrong implicit memory booking can result in unjustified OOM terminations.
Sizing review	If all above checks didn't help to reduce the OOM situations you should double-check the SAP HANA sizing. See SAP Note 2000003 ("What has to be considered for sizing SAP HANA?") for more information.

10. How can I judge if the available memory is sufficient for the current system and a projected future growth?

There are some general rules of thumb available that can help to understand if the memory is properly sized in an existing system, e.g.:

- Memory size should optimally be at least two times the total size of row store and column store.
- The memory used by SAP HANA should be significantly below the SAP HANA allocation limit (exception: Large caches that can be shrinked automatically on demand)

All these rules are only rough guidelines and there can always be exceptions. For example, lome large S/4HANA systems can work absolutely fine even if 65 % of the memory is populated with table data.

At this point we won't use these rules but instead describe a more detailed approach based on a real-life SAP Suite on HANA system with 4 TB of physical memory.

In a first step it is important to understand how much memory is allocated by the different main areas. This information is retrieved via *SQL: "HANA_Memory_TopConsumers" (DATA_SOURCE = 'CURRENT', AGGREGATE_BY = 'AREA')*:

```
---------------------------------------------- |AREA |SIZE_GB |SIZE_PCT|CUM_SIZE_PCT| ---------
------------------------------------- |Column store| 1011.72| 60.55| 60.55| |Heap area |
446.89| 26.74| 87.30| |Row store | 128.77| 7.70| 95.01| |Code | 6.62| 0.39| 95.41| |Stack |
1.58| 0.09| 95.50| ----------------------------------------------
```

We can see that around 1.1 TB are used by the column store, 0.1 TB is used by the row store and additional 0.4 TB are used by heap areas (that are not integral part of other areas). The total memory utilization of SAP HANA is significantly below 2 TB, so we can already conclude that there is a lot of safety margin for exceptional situations and future growth before the 4 TB memory limit is reached.

More detailed information can be determined with *SQL: "HANA_Memory_Overview"* (SAP Note 1969700). The output for the same system looks like:

```
------------------------------------------------------------------------------------------------
--------------------------------------------------------- |NAME |TOTAL_GB |DETAIL_GB |DETAIL2_GB

------------------------------------------------------------------------------------------------
--------------------------------------------------------- |User-defined global allocation
limit|not set | | | | | | | | |License memory limit | 4000| | | | | | | |License usage | 3000|
1554 (2014/03/01-2014/03/31)| | | | 2873 (2014/04/01-2014/04/30)| | | | 2849 (2014/05/01-
2014/05/31)| | | | 3000 (2014/06/01-2014/06/27)| | | | | | |Physical memory | 4040| 4040
(hlahana21) | | | | | | |HANA instance memory (allocated) | 3450| 3450 (hlahana21) |
| | | |
|HANA instance memory (used) | 1639| 1639 (hlahana21) |
| | | |
|HANA shared memory | 121| 121 (hlahana21) |
| | | |
|HANA heap memory (used) | 1508| 1508 (hlahana21) | 355 (Pool/NameIdMapping/RoDict)
| | | | 192 (Pool/AttributeEngine-IndexVector-Sp-Indirect)
| | | | 105 (Pool/AttributeEngine-IndexVector-Single) |
| | | | 102 (Pool/PersistenceManager/PersistentSpace(0)/DefaultLPA/Page)|
| | | | 85 (Pool/RowEngine/QueryExecution) |
| | | | 73 (Pool/AttributeEngine/idattribute) |
| | | | 66 (Pool/Statistics) |
| | | | 58 (Pool/AttributeEngine) |
| | | | 44 (Pool/AttributeEngine-IndexVector-SingleIndex) |
| | | | 38 (Pool/RowEngine/CpbTree) |
| | | | |
|Column store size | 1011| 1011 (hlahana21) | 315 (KONV) |
| | | | 84 (BSEG) |
| | | | 42 (ZARIXSD5) |
| | | | 36 (VBFA) |
| | | | 32 (ZARIXSD2) |
| | | | 31 (EDID4) |
| | | | 29 (BSIS) |
| | | | 28 (CDPOS) |
| | | | 25 (ZARIXMM2) |
| | | | 18 (KONP) |
| | | | |
|Row store size | 129| 129 (hlahana21) | 37 (A726) |
| | | | 30 (TST03) |
| | | | 12 (EDIDS) |
| | | | 7 (SRRELROLES) |
| | | | 5 (EDIDC) |
| | | | 4 (D010TAB) |
| | | | 4 (SWNCMONI) |
| | | | 3 (/SDF/MON) |
| | | | 3 (DD03L) |
| | | | 2 (REPOSRC) |
| | | | |
|Disk size | 1194| 1194 (global) | 320 (KONV) |
| | | | 104 (BSEG) |
| | | | 42 (ZARIXSD5) |
| | | | 36 (VBFA) |
| | | | 32 (ZARIXSD2) |
| | | | 30 (EDID4) |
| | | | 30 (TST03) |
| | | | 29 (BSIS) |
| | | | 27 (CDPOS) |
| | | | 25 (ZARIXMM2) |
------------------------------------------------------------------------------------------------
---------------------------------------------------------
```

The heap memory size is reported with 1508 GB which is much more than the 447 GB from further above. The reason is that in the second result list all heap areas are considered, also the ones that are the basis for the column store. This means, most of the 1508 GB heap allocation overlaps with the column store size. The shared memory size of 121 GB overlaps with the row store.

The allocated instance memory of 3450 GB is much higher than the used instance memory of 1639 GB, because SAP HANA tends to keep allocated memory allocated as long as there is no memory shortage. From a sizing perspective the used memory matters.

So also the memory overview output indicates that the used memory is significantly below 2 TB and far away from the 4 TB memory limitation.

A closer look into the top heap areas (*SQL: "HANA_Memory_TopConsumers", DATA_SOURCE = 'CURRENT', AREA= 'HEAP', AGGREGATE_BY = 'DETAIL'*) shows the following top allocators for the same system:

```
-------------------------------------------------------------------------------- |DETAIL
|SIZE_GB
--------------------------------------------------------------------------------
|Pool/PersistenceManager/PersistentSpace(0)/DefaultLPA/Page | 105.70|
|Pool/RowEngine/QueryExecution | 84.32
|Pool/Statistics | 65.97
|Pool/JoinEvaluator/TranslationTable | 24.90| ------------------------------------------------
---------------------------------
```

The Page allocator being responsible for a memory utilization of 106 GB is a kind of file system buffer that can reduce its size without problems whenever there is a memory shortage. So we can assume that another around 80 GB could be saved if required. This means that the total required memory is 1550 GB.

Conclusion: Even if the used memory size doubles it is still well below the memory limit (3100 GB vs. 4000 GB) and can also handle exceptional situations (e.g. significant growth of certain heap allocators) without running into memory pressure.

It is useful to repeat this analysis from time to time and also check the historic memory utilization (*SQL: "HANA_Memory_TopConsumers", DATA_SOURCE = 'HISTORY'*) to get a good understanding of the memory requirements over time.

11. Is it possible to monitor the memory consumption of SQL statements?

You can activate the statement memory tracking feature by setting the following parameters:

```
global.ini -> [resource_tracking] -> enable_tracking = on global.ini -> [resource_tracking] ->
memory_tracking = on
```

Changes to both parameters can be done online, no restart is required.

When memory tracking is active, the following memory information is available:

Patch level	Table	Column
>= Rev. 1.00.80	M_EXPENSIVE_STATEMENTS	MEMORY_SIZE
>= Rev. 1.00.94	M_ACTIVE_STATEMENTS M_PREPARED_STATEMENTS	ALLOCATED_MEMORY_SIZE USED_MEMORY_SIZE AVG_EXECUTION_MEMORY_SIZE MAX_EXECUTION_MEMORY_SIZE MIN_EXECUTION_MEMORY_SIZE TOTAL_EXECUTION_MEMORY_SIZE

>= Rev. 1.00.94 >= Rev. 1.00.100	M_CONNECTION_STATISTICS M_SQL_PLAN_CACHE	AVG_EXECUTION_MEMORY_SIZE MAX_EXECUTION_MEMORY_SIZE MIN_EXECUTION_MEMORY_SIZE TOTAL_EXECUTION_MEMORY_SIZE

Due to a bug with Rev. 1.00.90 to 1.00.96 (SAP Note 2164844) the setting will only work if additionally also the statement_memory_limit parameter (see below) is set.

Before Rev. 1.00.94 the expensive statement trace could only be triggered by runtimes of SQL statements. Starting with Rev. 1.00.94 you can use the following parameter to trigger the recording of expensive SQL statements in M_EXPENSIVE_STATEMENTS based on the memory consumption:

```
global.ini -> [expensive_statement] -> threshold_memory = <bytes>
```

12. Is it possible to limit the memory that can be allocated by a single SQL statement?

Starting with SAP HANA 1.0 SPS 08 you can limit the memory consumption of single SQL statements. As a prerequisite you need to have the statement memory tracking feature enabled as described above. Additionally you have to set the following parameter in order to define the maximum permitted memory allocation per SQL statement and host:

```
global.ini -> [memorymanager] -> statement_memory_limit = <maximum_memory_allocation_in_gb>
```

Starting with SAP HANA 2.0 SPS 00 you can define the amount of allocated memory per host for all concurrent database requests:

```
global.ini -> [memorymanager] -> total_statement_memory_limit =
<maximum_memory_allocation_in_gb>
```

For more details see SAP Note 2222250 ("How can workload management be configured for memory?").

13. What can I do if a certain heap allocator is unusually large?

See SAP Note 1840954 for some general advice.

The following table contains allocator-specific recommendations. Normally there is no need to perform manual analysis and optimization, so make sure that you are in a pathologic or critical situation before you consider any changes:

Allocator	Purpose	An
AllocateOnlyAllocator-limited/FLA-Li<40,64>/MinReadTSEntry	Minimum read timestamp tracker	This allocator is required to timestamp of a SAP HANA garbage collection purposes ordered list of read timestar purged as soon as it is no lo collection is no longer block Significant growth can happ blocked for a longer time. T reduces unless SAP HANA
AllocateOnlyAllocator-unlimited/FLA-UL<3145728,1>/MemoryMapLevel2Blocks (SAP HANA 1.0) AllocateOnlyAllocator-unlimited/FLA-UL<24592,1>/MemoryMapLevel3Nodes (SAP HANA >= 2.0)	Internal memory management	This allocator contains infor HANA memory. Normally no memory utilization or freque system can result in the allo which will result in a larger a consider the following optim

		defragmentations (hdbcons gc_unused_memory_thresh container shrinks (hdbcons unload_upper_bound param (not too small) thresholds) a of parameters async_free_t (SAP Note 2169283). Analy memory situation (e.g. in te intermediate results) in orde memory defragmentations c experience a very high (and consumption due to this allo with the following hdbcons c SAP HANA 1.0: mm level2r pagetable Large sizes of thi running into address space represents around 170 GB indications exist that an OO operating system?" below fo into address space related (following limits are reached size Intel no 768 GB Intel ye BIGMEM) no 96 GB IBM on IBM on Power (BIGMEM) n yes 256 GB With SAP HAN and >= 2.00.023 the growth minimum size of allocations operating system, e.g. to 32 [memorymanager] -> min_s must be used (i.e. 4, 8, 16, result in indexserver crashe below 4 (MB) must not be c increase the overhead when from the operating system, negative side effects. The c determined via M_SERVICE_MEMORY.ME >= 2.00.040). In the future S min_segment_size when a space is already consumed memory management is op of this allocators are less lik reduces unless SAP HANA
Pool/AdapterOperationCache	SDQ adapter operation cache	This heap allocator is used adapter operation cache. Se information about SAP HAN adapter operation cache in disabled via: scriptserver.in enable_adapter_operation_ HANA 2.0 SPS 03 the cach command: ALTER SYSTEM ('AdapterOperationsCache')

Pool/ASTRuleEngine Pool/ASTRuleEngine/ASTRuleEngine ExternalApi	SQL preprocessing cache for internal objects	This allocator is used in cor in order to store internal obj preprocessing. Large sizes like for allocator Pool/SQLP Check the Pool/SQLParser possible root causes and so
Pool/Auditing	Auditing	This allocator stores auditin 2159014). In case of large a disable auditing as a tempo 1.00.122.13 - 1.00.122.14 r allocator. This problem is fix 1.00.122.15. If you don't use sure that the following SAP because as long as this par overhead due to auditing is audited: global.ini -> [auditin global_auditing_state
Pool/BackupCopier/SynchronousPoolCopyHandler	Multistream channel copy	This allocator is used in con allocated size is linked to th parallelism: Number of para global.ini -> [backup] -> parallel_data_backup_back are only used when the volu defined with the following pa global.ini -> [backup] -> parallel_data_backup_back (default: 512 MB): global.ini data_backup_buffer_size T allocator size apply (with #s services that are backed up type Backup encryption Ma 4 * data_backup_buffer_siz data_backup_buffer_size * data_backup_buffer_size * * parallel_data_backup_bac data_backup_buffer_size * parallel_data_backup_back data_backup_buffer_size * parallel_data_backup_back data_backup_buffer_size * 2.00.045 you can set the fo eliminate the memory overh (SAP Note 2222250): globa enable_parallel_backup_en same size rules apply like fo Attention: Reducing the buf negative impact on backup
Pool/BWFlattenScenario	BW infocube conversion	This allocator is used during optimized cubes using BW_CONVERT_CLASSIC_ executed when a classic inf optimized infocube using tra

		Increased memory consum infocubes are converted. Af infocubes is finished, execu required and the allocator s
Pool/AttributeEngine/Delta/BtreeDictionary Pool/AttributeEngine/Delta/Cache Pool/AttributeEngine/Delta/InternalNodes Pool/AttributeEngine/Delta/LeafNodes Pool/ColumnStoreTables/Delta/BtreeDictionary Pool/ColumnStoreTables/Delta/Btreeindex Pool/ColumnStoreTables/Delta/Cache Pool/ColumnStoreTables/Delta/InternalNodes Pool/ColumnStoreTables/Delta/LeafNodes	Delta storage components	See SAP Note 2057046 and properly configured and exe size of the tables remains o
Pool/AttributeEngine Pool/AttributeEngine/idattribute Pool/AttributeEngine-IndexVector-BlockIndex Pool/AttributeEngine-IndexVector-BTreeIndex Pool/AttributeEngine-IndexVector-Single Pool/AttributeEngine-IndexVector-SingleIndex Pool/AttributeEngine-IndexVector-Sp-Cluster Pool/AttributeEngine-IndexVector-Sp-Indirect Pool/AttributeEngine-IndexVector-Sp-Prefix Pool/AttributeEngine-IndexVector-Sp-Rle Pool/AttributeEngine-IndexVector-Sp-Sparse Pool/ColumnStore/Main/Dictionary/RoDict Pool/ColumnStoreTables/Main/Compressed/Cluster Pool/ColumnStoreTables/Main/Compressed/Indirect Pool/ColumnStoreTables/Main/Compressed/Prefix Pool/ColumnStoreTables/Main/Compressed/Rle Pool/ColumnStoreTables/Main/Compressed/Sparse Pool/ColumnStoreTables/Main/Dictionary/RoDict Pool/ColumnStoreTables/Main/Dictionary/ValueDict Pool/ColumnStoreTables/Main/Index/Block Pool/ColumnStoreTables/Main/Index/PageableBlock Pool/ColumnStoreTables/Main/Index/PageableSingle Pool/ColumnStoreTables/Main/Index/Single Pool/ColumnStoreTables/Main/PagedUncompressed Pool/ColumnStoreTables/Main/Rowid Pool/ColumnStoreTables/Main/Text/DocObjects Pool/ColumnStoreTables/Main/Uncompressed Pool/NameIdMapping/RoDict	Column store components	These allocators are respor Their memory allocation wil amount of table data in colu reduction of indexes, ...). Se management suggestions c
Pool/ColumnStore/Main/Rowid/build-reverse-index Pool/ColumnStoreTables/Main/Rowid/build-reverse-index	Temporary structure when creating reverse index on $rowid$ column	This allocator is used during compression runs (SAP No is created on the $rowid$ co
Pool/AttributeEngine/Transient Pool/AttributeEngine/Transient/updateContainerConcat	Transient column store information	These allocators contain te can grow significantly in cas 2160391). The behavior is i 1.00.122.11 SAP HANA 2.0 consider creating indexes a are empty or filled to a mino
Pool/BackupRecoveryAllocator	Backup / recovery	This allocator stores inform

	information	recovery. A large backup ca increased allocator size (SA
Pool/BitVector	Basic data structure (e.g. temporary query results, columnar data, transactional info of column tables)	Can be linked to problems w store, see "Which options e memory issues?" -> "Transa
Pool/BWFlattenScenario	BW infocube conversion	This allocator is used during optimized cubes using BW_CONVERT_CLASSIC_ executed when a classic inf optimized infocube using tra Increased memory consum infocubes are converted. Af infocubes is finished, execu required and the allocator s
Pool/CacheMgr/CS_StatisticsCache	Column store statistics cache	This heap allocator is used (SAP Note 2502256). Starti cache can be cleared using SYSTEM CLEAR CACHE (
Pool/CacheMgr/DataStatisticsAdviserCache	Data statistics adviser cache	This heap allocator is used (SAP Note 2502256). Starti cache can be cleared using SYSTEM CLEAR CACHE (
Pool/CacheTransContainer	Transaction related cache invalidation information	This cache stores cache inv transactions. It can grow wh (SAP Note 2169283) or whe transactions or holdable cu
Pool/CalculationEngine	Calculation engine structures	This heap allocator contains structures, so most of M_CE_CALCSCENARIOS. addition it is also a parent a plans.
Pool/ChannelUtils/SynchronousPoolCopyHandler	Multistream channel copy	This is an earlier name for a Pool/BackupCopier/Synchro see the Pool/BackupCopier section above.
Pool/AttributeEngine/Delta Pool/ColumnStore/Delta Pool/ColumnStoreTables/Delta	Temporary delta storage related data	Unlike other Pool/.../Delta* a delta related data only to a massively during ongoing c can be considered more as allocator. In order to reduce reduce the amount of recor modification command. A b <= 2.00.024.07 and <= 2.00 increase of these allocators chunk-wise data processing
Pool/AttributeEngine/Delta/BtreeDictionary Pool/AttributeEngine/Delta/Cache	Delta storage component	See SAP Note 2057046 and properly configured and exe

Pool/AttributeEngine/Delta/InternalNodes Pool/AttributeEngine/Delta/LeafNodes Pool/ColumnStore/Delta/BtreeDictionary Pool/ColumnStore/Delta/Btreeindex Pool/ColumnStore/Delta/Cache Pool/ColumnStore/Delta/InternalNodes Pool/ColumnStore/Delta/LeafNodes Pool/ColumnStoreTables/Delta/BtreeDictionary Pool/ColumnStoreTables/Delta/Btreeindex Pool/ColumnStoreTables/Delta/Cache Pool/ColumnStoreTables/Delta/InternalNodes Pool/ColumnStoreTables/Delta/LeafNodes		size of the tables remains o
Pool/AttributeEngine Pool/AttributeEngine/idattribute Pool/AttributeEngine-IndexVector-BlockIndex Pool/AttributeEngine-IndexVector-BTreeIndex Pool/AttributeEngine-IndexVector-Single Pool/AttributeEngine-IndexVector-SingleIndex Pool/AttributeEngine-IndexVector-Sp-Cluster Pool/AttributeEngine-IndexVector-Sp-Indirect Pool/AttributeEngine-IndexVector-Sp-Prefix Pool/AttributeEngine-IndexVector-Sp-Rle Pool/AttributeEngine-IndexVector-Sp-Sparse Pool/ColumnStore/Main/Compressed/Cluster Pool/ColumnStore/Main/Compressed/Indirect Pool/ColumnStore/Main/Compressed/Prefix Pool/ColumnStore/Main/Compressed/Rle Pool/ColumnStore/Main/Compressed/Sparse Pool/ColumnStore/Main/Dictionary/RoDict Pool/ColumnStore/Main/Dictionary/ValueDict Pool/ColumnStore/Main/Index/Block Pool/ColumnStore/Main/Index/Single Pool/ColumnStore/Main/PagedUncompressed Pool/ColumnStore/Main/Rowid Pool/ColumnStore/Main/Text/DocObjects Pool/ColumnStore/Main/Uncompressed Pool/ColumnStoreTables/Main/Compressed/Cluster Pool/ColumnStoreTables/Main/Compressed/Indirect Pool/ColumnStoreTables/Main/Compressed/Prefix Pool/ColumnStoreTables/Main/Compressed/Rle Pool/ColumnStoreTables/Main/Compressed/Sparse Pool/ColumnStoreTables/Main/Dictionary/RoDict Pool/ColumnStoreTables/Main/Dictionary/ValueDict Pool/ColumnStoreTables/Main/Index/Block Pool/ColumnStoreTables/Main/Index/Single Pool/ColumnStoreTables/Main/PagedUncompressed Pool/ColumnStoreTables/Main/Rowid Pool/ColumnStoreTables/Main/Text/DocObjects Pool/ColumnStoreTables/Main/Uncompressed Pool/NameIdMapping/RoDict	Main storage components	These allocators are respor in column store. Their mem you reduce the amount of ta cleanup, reduction of indexe
Pool/ColumnStore/System	Column store metadata	This allocator contains colu mainly depends on the num columns. Sizes up to 10 GE

		100,000 to 150,000 tables. relative allocator size can b table sizes. This is expectec
Pool/commlibDefAllocator Pool/ncCommLibDefAllocator	Network communication support objects	This allocator can grow in c channels exist (due to inter- communication), see SAP N related to SAP HANA netwo with SAP HANA 1.00.122.0 with SAP HANA 2.00.021.
Pool/Contexts	Context information	This heap allocator stores in contexts. An increased size number of contexts. For mo related information for heap
Pool/Crypto	Encryption related data structures	This allocator can grow with due to a memory leak relate HASH_SHA256). You need reclaim the allocated memo (that can be checked via an described in SAP Note 222 Crypto::Provider::Common(
Pool/CS/BufferPage	NSE buffer cache	This heap allocator is used (NSE) buffer cache (SAP N can be controlled with the fo -> [buffer_cache_cs] -> max indexserver.ini -> [buffer_ca <max_percent_of_memory>
Pool/CSPlanExecutor/PlanExecution	Intermediate data structures	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al execution of a database rec cases where specific statem
Pool/CSRowLocking	Column store row locking	This allocator is typically lar can check the current recor "HANA_Locks_Transaction The allocator is purged asyn (SAP Note 2057046) and co Only an unload guarantees are completely purged. The allocations to specific tables modifications it is normal th remain at a certain size alth locks. In general this does n bottleneck. In case of perma more detailed analysis is us SPS 04 the row locks are cl shouldn't remain on large si HANA 2.0 SPS 04 it is also row locking details including "HANA_Locks_Transaction Note 1969700).

Pool/CS_TableSearch	Query optimizer related data structures	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible.
Pool/DeletedPageList	Recording of DELETE operations in row store	This allocator is typically qu problem described in SAP N - 102.02) it can already bec >= 2 MB.
Pool/DocidValueArray	Set of rowids and related values in context of join engine	See question "Which gener the SQL statement memory make sure that join SQL sta efficient as possible.
Pool/DPServerFramework Pool/DPServerStatsRequestIfacerAllocator	Data provisioning server memory	These allocators are used b (dpserver) that is used in sm (SAP Note 2400022). The f and rising allocator sizes ex leak accessing remote clust 1.00.122.12, <= 2.00.012.0 Risk of increased memory r (SAP HANA <= 1.00.122.19 SAP Note 2749005: Memor capture (CDC)
Pool/DSO/DSORead Pool/DSO/DSOUpdate	DSO activation / rollback	These allocators temporaril in BW like DSO_ACTIVATE DSO_ROLLBACK_PERSIS and DSO_ROLLBACK_CHA finished, the majority of the
Pool/DynamicCachedView Pool/DynamicCachedView/ViewMatching	Dynamic result cache information	These allocators contain inf result cache (SAP Note 250
Pool/entityCache	MDX entity cache	This allocator contains MDX use SQL: "HANA_Memory_ "HANA_Memory_Caches_E CACHE_NAME = 'MdxEntit details about the current uti time of entries in the entity c execution of the underlying maintained independently. T HANA resource container th gets scarce. It is unloaded w so that a large cache size is Note 2502256 for more info caches. Starting with SAP H cleared using the following CLEAR CACHE ('MdxEntity
Pool/ESX	ESX runtime data	See question "Which gener the SQL statement memory make sure that join SQL sta efficient as possible. The Ex Note 2599949) uses Pool/E following scenarios can cau

		consumption: SAP HANA 2 context of PlanViz executio 2.00.000 - 2.00.023: Large UNION ALL (SAP Note 274
Pool/Exception	Exception related data	This allocator is used in co use SQL: "HANA_Threads_ThreadSa (LOCK_NAME = 'throwHelp 'HASH') to identify databas of exception handling.
Pool/ExecutorPlanExecution	Intermediate result sets	See question "Which gener the SQL statement memory make sure that join SQL sta efficient as possible.
Pool/FemsCompression/CompositeFemsCompression	FEMS compression	The form element selection BW queries with execution the amount of data transfer SAP HANA. In some cases memory requirements. See Execution Mode for more in execution modes. As a loca executing the query in ques acceptable alternative. Also worth a try, because the un different and may not run th workaround you can disabl _GET_TREX_REQ_FLAGS CL_RSDRV_TREX_API_S line with a leading '*' (see p r_trex_req_flags = r_trex_re lead to disadvantages in oth transmitted data), you shou understood and fixed the re memory consumption.
Pool/Filter	intermediate result sets	See question "Which gener the SQL statement memory make sure that join SQL sta efficient as possible. This a e.g.: Pruning (join engine, C TRexApiSearch) Hierarchy memory should be release is finished.
Pool/FRSWLockAllocator	Read write locks	Starting with SAP HANA 1. locks are no longer stored i instead they are maintaine Pool/FRSWLockAllocator (" In case of a large size you via SQL: "HANA_Locks_Int (LOCK_TYPE = 'READWR 1969700. With SAP HANA allocator is reduced compa

		reduction is introduced with the granularity level was rec following known issues with exist: SAP HANA <= 1.00.1 SAP HANA <= 1.00.122.17 PreferredRoutingLocations information related to SAP I "HANA_Locks_Internal_Loc M_READWRITELOCKS do can explain the growth of th caused by unused (and the case you can use the follow 2222218) to dump the exist rw_out.txt for later analysis: command can generate a h to reduce the output to the i related counts you can use rwlocks.pl with the following my (%word, $key); while(<> $word{$key}) { $word{$key} foreach $key (keys(%word) Now you can run hdbcons a locks: hdbcons 'readwritelo awk -F\('{print $1}' |awk -F: 20 A large and rising numbe caused by blocked garbage DeltaRowIDReadWriteLock
Pool/hierarchyBlob Pool/hierarchiesItab	Hierarchy cache, MDX hierarchy cache	These allocators contain hie cache information that is po HANA views with hierarchie the core index, the Itab allo can use SQL: "HANA_Mem "HANA_Memory_Caches_E CACHE_NAME = '%Hierarc understand details about th cache. The life time of entri depend on the execution of they are maintained indepe of the SAP HANA resource the memory gets scarce. It than columns so that a larg critical. If hierarchies / cach by setting cache=false in th Enablement' = ' ' (instead of globally by disabling it with indexserver.ini -> [cache] -> hierarchies_transactional_c Note 2502256 for more info caches. Starting with SAP H cleared using the following CLEAR CACHE ('Hierarchy CACHE ('HierarchyCacheN

		CLEAR CACHE ('Hierarchy CLEAR CACHE ('MDXHiera
Pool/HierarchyFunctionsGeneralExec Pool/HierarchyFunctionsIncrementalLoad Pool/HierarchyFunctionsSingleton	Cache for SQL based hierarchies	These allocators are used f hierarchies. See SAP Note related to SAP HANA cache SPS 03 the cache can be c command: ALTER SYSTEM ('HierarchySqlFunctionCach
Pool/ICT	Internet communication toolkit allocations	The internet communication context of http requests via be caused by a memory lea 1.00.122.04 (SAP Note 239 specific bug the following S xsengine.ini -> [httpserver]
Pool/IndexRebuildAllocator	Memory area for row store index rebuilds	This issue can happen with 08. See SAP Note 2005478 a workaround in order to dis during startup: indexserver. use_jobex_index_rebuild =
Pool/IndexVector Pool/IndexVectorAligned	Temporary index vector structures	This allocator is used in diff optimizations, column load, creation. A temporary large merges (SAP Note 205704 (SAP Note 1871386), e.g. in Note 2416490).
Pool/itab	Column store (intermediate) search results, MDX hierarchy cache	Pool/itab can be used for di result sets (not part of resou statement execution) MDX container, released in conte when MDX hierarchy cache engine node cache (part of context of resource containe like planning sessions are c (SAP Note 2800007) There of Pool/itab corresponds to sizes (SAP Note 2502256) size and growth can be norm memory leak / memory rele Pool/itab allocator in contex often a consequence of the corresponding to custom vi cache. This is currently also columns and this behavior c memory footprint. Another i size of table CS_VIEW_ATT hierarchy cache and the rel automatically during resour gets scarce. A manual redu be achieved in the following Resource container shrink (hdbcons or unload_upper_b

		2.0: ALTER SYSTEM CLEA Configure parameter indexs cache_entry_timeout / cach described in SAP Note 250 entries are marked as stale large temporary column tab can use SQL: "HANA_Tabl (IS_COLUMN_TABLE = 'YE via SAP Note 1969700. The problem scenarios that can size of Pool/itab: SAP Note leak calling virtual procedur integration (SDI, SAP Note tables or bugs aren't respon check question "Which gen the SQL statement memory make sure that SQL statem efficient as possible. If no e Pool/itab allocator is found, as described in SAP Note 2
Pool/itab/VectorColumn	Column store (intermediate) search results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. If no e Pool/itab allocator is found, as described in SAP Note 2
Pool/JERequestHandler	Temporary structure during translation table creation	This allocator is required in tables are created to suppo 1998599 for more informati
Pool/JoinEvaluator	Global join engine allocator	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This is should normally not be used Instead sub-allocators are u please check if also some s If yes, proceed with the ana Pool/JoinEvaluator/* sub al Note 2370588 describes a size for Pool/JoinEvaluator, allocator Pool/JoinEvaluato large. A large size of Pool/J with SAP HANA Rev. 122.0 access (FDA), so you can c ALL ENTRIES as described rsdb/prefer_join_with_fda a 0. A large size of Pool/Join of a large query result cach
Pool/JoinEvaluator/DictsAndDocs	Join engine dictionaries	See question "Which gener the SQL statement memory make sure that SQL statem

		efficient as possible. This al HANA SQL statement proce AttributeEngine::AttributeAp JoinEvaluator::JEDistinctAtt . Among others the followin significant growth of this allo partitioned tables are respo partitioning. Check for COU (partitioned) tables and colu (SAP Note 2000002 -> "Wh expensive SQL statements DISTINCT"). This allocator created because the join en processing.
Pool/JoinEvaluator/IndexInfo	Join engine cache	This allocator stores informa or join statistics in a specific A large size is typically linke partitions or columns.
Pool/JoinEvaluator/JECalculate Pool/JoinEvaluator/JECalculate/TmpResults Pool/JoinEvaluator/JECreateNTuple Pool/JoinEvaluator/JEPreAggregate Pool/JoinEvaluator/JEStep1 Pool/JoinEvaluator/JEStep2 Pool/JoinEvaluator/NTuple	Join engine intermediate data structures	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. If you statements responsible for SQL: "HANA_Threads_Thre (THREAD_DETAIL = '%(JE THREAD_DETAIL') availab for SQL statements with a s join engine functions. Consi USE_OLAP_PLAN (SAP No (SAP Note 2570371) for tes switch from join engine to O reduced memory consumpt Pool/JoinEvaluator/JECreat joins (e.g. EXCEPT) and ca JoinEvaluator::LoopJob::fin by a SAP HANA bug that is 2.00.010. With SAP HANA default. With SAP HANA <= default and can be activated CONSERVATIVE_CS_ANT with the following paramete conservative_cs_anti_join_ workaround the NO_GROU Note 2142945) can be used also disable the RSADMIN (SAP Note 1865554). Large Pool/JoinEvaluator/JECreat Pool/JoinEvaluator/NTuple tables (SAP Note 2340450) with SAP HANA >= 2.00.03
Pool/JoinEvaluator/JEEvalPrecond	Join engine intermediate data	See question "Which gener the SQL statement memory

	structures and metadata	make sure that SQL statem efficient as possible. This al metadata (modules TRexConfig::CachedMetaD cessors, TRexConfig::CachedMetaD , TRexConfig::CachedMeta on, TRexConfig::CachedMe whole life time of the statem a large SQL cache and man engine it is possible that this remains at this level. Clearin reduce the allocator size. S information related to parsin
Pool/JoinEvaluator/JEPlanData/deserialized	Join engine intermediate data structures involving inter-node communication	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Check across nodes is already opt less inter-node data transfe 2081591 for more informatio distribution. Consider settin Note 2142945) for testing p from join engine to OLAP en reduced memory consumpt
Pool/JoinEvaluator/JEAggregate Pool/JoinEvaluator/JEAggregate/Results Pool/JoinEvaluator/JEAssembleResults Pool/JoinEvaluator/JEAssembleResults/Results Pool/JoinEvaluator/JECalculate/Results Pool/JoinEvaluator/JERequestedAttributes/Results	Join engine results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. If you c statements responsible for t SQL: "HANA_Threads_Thre (THREAD_DETAIL = '%(JE THREAD_DETAIL') availab for SQL statements with a s join engine functions. This a when late materialization isn bugs described in SAP Note parameters may be increas requirements: indexserver.i late_materialization_thresho late_materialization_thresho parameters as soon as you a revision level with include USE_OLAP_PLAN (SAP No (SAP Note 2570371) for tes switch from join engine to O reduced memory consumpt memory consumption are: S HANA Rev. 1.00.91, fixed w 2260972 (inappropriate imp procedures) SAP Note 2370 S/4HANA migration routines

		missing calc view unfolding decimal notation (fixed with and 2.00.024)
Pool/JoinEvaluator/PlanDataAttrVals/Deserialized	Join engine results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al results have to be sent from scenarios.
Pool/JoinEvaluator/TranslationTable	Join column mapping	Translation tables are requi values. SAP Note 1998599 configured in order to optim scenarios a significant mem of translation tables related SAP HANA Rev. 102.02 tra table joins are no longer ke Rev. 97.02 and higher you indexserver.ini -> [joins] -> 'false' See SAP Note 2217
Pool/JoinEvaluator/ValueList	Intermediate join engine value list	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. An inc migration can be a consequ Note 2370588.
Pool/KernelSentinel	Kernel sentinel	The kernel sentinel takes o previous MVCC anti ager in describes a problem with S can result in an increased a
Pool/L/jit/CodeCache	Cache for compiled L code	This allocator contains com programs are compiled in c You can use SQL: "HANA_ 1969700) to check how the with a mid-term disposition issues and resource contai evict cache entries. A large LlangGlobalCodeMap_Writ contention on SAP HANA <
Pool/L/jit/MetaData Pool/L/llang/Debuggee	Intermediate Llang structures (for compiled programs / for interpreting and debugging)	These heap allocators cont HANA <= 2.00.023 the Poo part of the SAP HANA reso shrinked in case of low me critical in context of the SAP activation (SAP Note 25703 when many parsed queries and so the allocator size ca 2.00.024 this problem is fixe the resource container so th is required. As a workaroun can set the following SAP H [execution] -> compilation_

		[execution] -> asynchronou SQL cache (SAP Note 2124 reduce the allocator size wi SYSTEM CLEAR SQL PLA the SQL cache will result in and so it should only be per With SAP HANA >= 2.00.03 context of HEX queries. Thi because the allocator is par data will be evicted in case container shrinks. If require unload_upper_bound. With recommended to disable th 2600030) to avoid the grow enable_interpreter_cache = more details.
Pool/L/llang/CodeCache	Llang structures (for compiled programs)	Pool/L/llang/CodeCache is coming from SqlScript, grap mid-term disposition weight resource container shrinks S entries. A large cache can b LlangGlobalCodeMap_Writ contention on SAP HANA <
Pool/L/llang/Runtime/Global Pool/L/llang/Runtime/Local	Intermediate Llang script results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Both a intermediate Llang script re statement is finished. The L single thread (faster, copy o needs to access it) while th by different threads (typicall arrays). Llang queries may of FOX, SQLScript or HEX Pool/L/llang/Runtime/Local strings or CLOB values is p program can be found in the _SYS_PLE:2016012611442 following case: 20: 0x00007 ljit::dynamic/_split_main_0+ fox/cen_"SAPSR3"."_SYS_ TMPDATA". cv053_fox_LLangView.56A 52:0 (<unknown>) If require ljit component on debug lev indexserver.ini -> [trace] ->
Pool/LVCAllocator/LVCContainerDir Pool/LVCAllocator/LVCContainerDir/LVCContainer_<id > Pool/LVCAllocator/LVCObjectPageDir Pool/LVCAllocator/LVC_ObjectPageDir	liveCache data	These allocators hold the a sizes should correspond to SAP Note 2593571 for more
Pool/LVCAllocator/OMSAllocator/Session_<conn_id>/O	liveCache changes and	This allocator stored change

MSSession/OMSDefaultContext	processed objects	transaction is finished. Duri persisted and the space is r allocator size indicates that liveCache data exist or that to be processed.
Pool/LVCAllocator/OMSAllocator/Session_<conn_id>/USERSession/OMS User COMRoutine	liveCache intermediate procedure data	This allocator is used for da liveCache procedures (e.g. <conn_id> in the name refe responsible for the procedu memory is released once th memory requirements are a data volume and analysis is SAP Note 2593571 for mor
Pool/malloc/hdbindexserver	General indexserver allocator	This allocator can grow sign handling. You can use SQL "HANA_Threads_ThreadSa (LOCK_NAME = 'throwHelp 'HASH') to identify database of exception handling.
Pool/malloc/libdbrsa16_r.so Pool/malloc/libdbrsa17_r.so	Sybase IQ remote accesses	These libraries are used for IQ remote accesses. In SAP are usually related to smart 2180119). There is a memo accesses with Open SSL an with Sybase IQ levels >= 16 11921 and >= 16.1 SP01 12
Pool/malloc/hdbnameserver	Temporary nameserver data structures	This allocator is used for ter that are e.g. used in contex actions like a full system inf performance trace (SAP No
Pool/malloc/libc.so.6	Linux libc allocations	This allocator is used when Linux libc.so library, e.g. in t like __alloc_dir and System allocator in the context of an assume that it is only a victi for example the compileserv memory for Pool/malloc/libc indexserver at first, because available memory before.
Pool/malloc/libhdbbasement.so	Column store data structures	A large and growing size ca reasons: A memory leak in result in growth of this alloc TrexThreads::InheritableLoc is fixed with SAP HANA 1.0 consistency check contexts function profiler with SAP H 2.00.024.01 and 2.00.030 c (SAP Note 2637828).
Pool/malloc/libhdbcalcengine.so Pool/malloc/libhdbcalcengineapi.so	Calculation engine intermediate results	See question "Which gener the SQL statement memory

		make sure that SQL statem efficient as possible. The fo responsible for increased al growth of this allocator in co processing (e.g. TREXviaD TrexCalculationEngine::Opt yJoinOverMultiprovider and TrexCalculationEngine::Co e::applyAggrOverHierachyJ that is fixed with SAP HANA 2374935 describes a memo scenario modeler objects th 112.07 and 122.04.
Pool/malloc/libhdbcalcenginepops.so	Intermediate results during calculation engine plan operation processing	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. In mos the main contributor for the can use calculation view un CALC_VIEW_UNFOLDING
Pool/malloc/libhdbcs.so	Column store components	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. If you s allocator, check the followin being created on a partition significant amounts of mem can avoid creating indexes column store tables. If the p activity, see SAP Notes 225 that critical indexes are crea Starting with SAP HANA 1. memory footprint of index c Pool/malloc/libhdbcs.so ma tables (see SAP Note 2057 on large tables and columns be responsible for a growth can happen in context of ca ceProjectionPop, call stack AttributeEngine::CachedEx ache::InternalCache, TrexCalculationEngine::ceF alculatedColumn). With mo heap allocator Pool/itab/exp context of pattern searches SAP HANA <= 2.00.037.00 you may observe call stack AttributeEngine::Delta::tree AttributeEngine::Delta::com allocator stack trace (SAP N Pool/malloc/libhdbcs.so is a query result cache (SAP No

Pool/malloc/libhdbcsaccessstatisticscache.so	Allocations related to access statistics cache	This allocator contains alloc table access statistics cach information how to configure
Pool/malloc/libhdbcsapi.so	Column store API (search) and intermediate results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Additio following specific constellati allocator in combination wit (e.g. TREXviaDBSL, call sta TrexCalculationEngine::Opt yJoinOverMultiprovider and TrexCalculationEngine::Cor e::applyAggrOverHierachyJ that is fixed with SAP HANA With SAP HANA <= 1.0 SP can be caused by the creati are dynamically created dur join after a restart of SAP H concurrently started in man and memory requirements a transaction calculates the jo workaround you can execut running it with a higher para 09 the join statistics creation longer happens. This alloca large delta storages are acc TRexAPI::DeltaIndexManag this scenario. In this case yo reasonable merge strategy 2057046). Large allocations expensive fuzzy / text searc modules like ltt_adp::vector TRexAPI::FreeStyleExecuto TRexAPI::FreeStyleExecuto TRexAPI::FreeStyleExecuto from module OlapEngine::B in context of BW multiprovic FEMS can be caused by mi with a convex hull optimizat See SAP Note 2517443 and indexserver.ini -> [calcengin optimize_convex_hull_throu With SAP HANA <= 1.00.12 2.00.024 a memory leak in searches (e.g. Enterprise S allocator size (SAP Note 26 Pool/malloc/libhdbcs.so is a query result cache (SAP No
Pool/malloc/libhdbcsmd.so	Transient metadata	This allocator contains trans merge time, column informa main and delta storage. Add

		HANA engines for specific r origination of the space con list or allocator stack trace o Note 2222218). Known sce Details TRexConfig::Attribu inition This module indicate column information. It is pop with specific SAP HANA en time is linked to the entry in optimizations exist: Avoid ta columns (> 1000) that are a requests Use the HEX engi longer populates this alloca (SAP Note 2142945). Clear 2124112) can temporarily re IGNORE_PLAN_CACHE (S SQL cache.
Pool/malloc/libhdbcsstore.so	Column store persistence objects	This allocator contains adm (like parts of the row lock in and may grow in case of ma collection. If much memory TrexStore::LockMapEntry*, row lock link hashmap garb you can trigger this garbage reloading tables with a high problem is fixed with SAP H
Pool/malloc/libhdbcstypes.so	Column store data types, hybrid LOB information	This allocator contains infor types including hybrid LOB values or disk LOB referenc common to see sizes betwe databases, depending on th existing in the system. This like the Pool/ColumnStoreT Its size is closely linked to th focus on hybrid LOB colum by reducing data stored in h 2220627 for more informatio
Pool/malloc/libhdbcswrapper.so	(Intermediate) results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. The fo increased memory footprint SAP HANA <= 1.00.122.16 inverted index joins (SAP N
Pool/malloc/libhdbevaluator.so	Intermediate results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Databa growth of this allocator typic modules like Evaluator::Thr
Pool/malloc/libhdbitab.so	Intermediate results	See question "Which gener

		the SQL statement memory make sure that SQL statem efficient as possible. This al processing similarly to Pool Pool/malloc/libhdbitab.so sh allocations should happen f growing, a memory leak can 1.00.97.02 and Rev. 1.00.1
Pool/malloc/libhdbmetadataobject.so	Metadata	This allocator is linked to th large and rising size can be fixed with SAP HANA >= 1. clearing the SQL cache with PLAN CACHE" can help to and reduce the allocator siz
Pool/malloc/libhdbolap.so	Intermediate OLAP engine results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Large context of large temporary B tables (0BW:CRM:BI0_0M* memory leak in this allocato 1.00.112.07 and 1.00.120.
Pool/malloc/libhdbsqlparser.so	SQL parsing default allocator	This heap allocator is the ge allocations.
Pool/malloc/libhdbpartitioning.so	Intermediate results in context of partitioned tables	This generic allocator can g partitioned tables. If you exp large sizes, open a SAP inc
Pool/malloc/libhdbpythonbase.so	Python initialization and execution	This allocator is related to a executing Python functions. pythontrace was enabled fo Nameserver ran out of mem Pool/malloc/libhdbpythonba
Pool/malloc/libhdbrskernel.so	Row store components	This allocator contains all d allocations which aren't ass With newer revisions the uti reduce. You can use the op additionally create an alloca Note 2222218) to determine allocator. The following indi exist: Top consumer SAP N ptime::Proc_insert_parallel: by ptime::Proc_insert_paral memory leak bug which is f ptime::RowInsertReproduci With SAP HANA Revisions allocator can grow due to a setting the following parame indexserver.ini -> [row_engi dynamic_parallel_insert_ma ptime::codegen_qp2so::gen Revisions 100 to 102.01 an

		ptime::codegen_qp2so::ger described in SAP Note 2275 ltt::char_traits<char> >::enla ptime::ServiceThreadSamp sample details are collected running, the allocator can g second a new copies are cr Pool/malloc/libhdbrskernel.s you should search for extre avoid or reduce them as mu workaround you can also di sample details: global.ini -> service_thread_sampling_m led = false Be aware that th of the system and so it shou permanent basis. This prob and 2.00.001, then only the collected. See also SAP No issues with particularly large consider a reduction of the you experience a large and open a SAP incident for cla
Pool/malloc/libhdbtableconsistencycheck.so	Table consistency check	This allocator is related to th CHECK_TABLE_CONSIST You can limit the number of tables or run it at times with to reduce the risk of critical
Pool/malloc/libsapcrypto.so	Encryption related data structures	This allocator can grow with due to a memory leak relate HASH_SHA256). You need reclaim the allocated memo HANA 2.00.022.
Pool/M_CONSISTENT_VIEW_STATISTICS	Consistent view details	This heap allocator stores c calling monitoring view M_C The size can be increased i collection (SAP Note 21692
Pool/mds	MDS cache (intermediate) result sets of InA / MDS queries	This allocator contains both results in context of MDS re SAP Note 2502256 for more about the MDS cache. If the the cache size or you see te dominated by statement exe general optimizations exist memory requirements?" bel database requests are exec possible. The memory is on query is executed. You can reducing the amount of proc
Pool/mds/CubeAxis	MDS axis data of result set or cube	See question "Which gener the SQL statement memory make sure that database re

		efficient as possible. The m / MDS query is executed. Y by reducing the amount of p SAP Note 2670064 for more
Pool/mdx	MDX query allocations	As of SAP HANA 1.0 SPS 0 memory allocation of Pool/m
Pool/Metadata/MetadataCache/MetadataGlobalCacheSlot	Metadata cache	The metadata cache allocat 1.0 SPS 12 and is used to s has to be retrieved from a re significantly if many DDL op DDL operations invalidate e increased sizes are: Blocke 2169283) A problem exists 1.00.122.03 that can result This problem is fixed with R cache entries if possible). B metadata information was s so the allocator Pool/RowEn workaround in case of large clear the it manually: ALTE CACHE This command will node where you are current HANA 1.00.122.13 this com on all SAP HANA nodes an if you want to clear only the service.
Pool/Metadata/SessionLocalItabContainer	Temporary table information	This heap allocator exists w higher and is used to store local data and session local temporary tables Session lo If you face a high size of thi M_TEMPORARY_TABLES "HANA_Tables_Temporary 1969700). Reasons for incr amount of temporary tables and 2.00.000 the session lo amounts of memory than re 1.00.122.06 and 2.00.001. S details.
Pool/NetworkChannelCompletionHandler	Network channel completion interface	This allocator holds network number of channels can inc following options exist to op Notes 2222200 and 238242 HANA network configuration high amount of partitions th required network channels
Pool/OptimizeCompression/<schema>:<table> Pool/OptimizeCompression/<schema>:_SYS_SPLIT_<table>~<partition>	Compression optimization	Allocators starting with Poo during compression optimiz 2112604 and make sure tha a reasonable way. Furtherm size of individual tables / pa

		level. Sizes above 50 GB sl
Pool/parallel Pool/parallel/aggregates Pool/parallel/align Pool/parallel/compactcol Pool/parallel/ihm Pool/parallel/pop Pool/parallel/temp_aggregates Pool/parallel/temp_dimensions Pool/parallel/temp_other	OLAP aggregation results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Large Pool/parallel/aggregates ha large temporary BW tables (0BW:CRM:BI0_0M*) or BW a high memory consumptio mode D in BW environment correction available via SAF the hint NO_USE_OLAP_P testing purposes in order to engine to join engine works consumption. If the issue ap problem improves using a c See BW on HANA and the information related to BW q is linked to SAP ABAP quer you can consider deactivati 2399993. If you face a high DTP activities in BW, you ca possible optimizations.
Pool/PersistenceLayer	Persistence information	This allocator holds some g information. The following r known: Memory leak with S 2.00.037.02 and <= 2.00.04 accesses (SAP Note 28431
Pool/PersistenceManager/Backup	Sorted list of page numbers for data backups	This allocator is populated a order to have a sorted list o streaming. Once the data b released. The size of the al case of large databases and (SAP Note 2220627).
Pool/PersistenceManager/Backup/Superblock	Backup superblock information	This allocator is used in cor replication activities. Itcan g 2.00.040 - 2.00.041 in case Note 2818480).
Pool/PersistenceManager/ContainerFileIDMapping	LOB container mapping	This allocator maps LOB co is particularly large, the follo amount of LOB data (can e. "HANA_Tables_LargestTab 1969700) Unnecessarily hig improved with Revision 1.00
Pool/PersistenceManager/DisasterRecoveryPrimary	Asynchronous system replication buffer	The main contributor to the asynchronous system replic significant size of asynchro it closely depends on the va <service>.ini -> [system_rep logshipping_async_buffer_s increase this buffer, you sho

		high redo log generation, ty (<service>.ini = indexserver global.ini technically also w increased space is allocated services), and so memory i 1.00.122.17 and 2.00.024.0 increased from 64 MB to 25 Note 2678164). See SAP N related to SAP HANA syste
Pool/PersistenceManager/DisasterRecoverySecondary	System replication related allocations on secondary site	The size of this cache is ma following system replication replication site (SAP Note 1 [system_replication] -> logshipping_replay_push_p default value of 20 relates t 2409671 for more informati
Pool/PersistenceManager/DisasterRecoverySecondary/ReplayLogCache	System replication log replay cache	This cache is used in syster Note 1999880) with a log re logreplay or logreplay_read performance by avoiding dis for the indexserver and nam services. Normally no chang special situations (e.g. as w the size can be adjusted wit <service>.ini -> [system_rep logshipping_replay_logbuffe
Pool/PersistenceManager/LOBContainerDirectory	Hybrid LOB directory	This allocator contains infor stored on disk (see SAP No mainly on the amount of hyl can grow in case of problem SAP Note 2169283 for more collection.
Pool/PersistenceManager/LogRecovery	Log recovery	This allocator is used to buf memory during recovery. Tl can be checked with SQL: ' Note 1969700). In case of a expect a memory allocation
Pool/PersistenceManager/MidSizeLOBContainerFileIDMapping Pool/PersistenceManager/MidSizeLOBContainerFileIDMapping/BackMap Pool/PersistenceManager/MidSizeLOBContainerFileIDMapping/EidMap Pool/PersistenceManager/MidSizeLOBContainerFileIDMapping/OwnerBackMap	Packed LOB metadata structures	These heap allocators are u metadata. They can becom are stored in packed LOBs. related information and con reduce the memory sizes: M e.g. by doing cleanup or arc Avoid memory thresholds o [sql] -> lob_memory_thresh packed LOBs the memory c administration structures ca actual LOB size.
Pool/PersistenceManager/PersistentSpace/DefaultConverter/ConvPage	Persistence metadata	This allocator holds persiste grow in case of a growing p files. See SAP Note 24000

		SAP HANA persistence.
Pool/PersistenceManager/PersistentSpace/DefaultLPA	Parent page allocator	This is the parent allocator f mentioned below like: Pool/PersistenceManager/P /DataPage Pool/PersistenceManager/P /LOBPage Pool/PersistenceManager/P /Page Pool/PersistenceMan /ShadowPage A large size c consequence of an even lar allocators. You have to ana child allocator in this case.
Pool/PersistenceManager/PersistentSpace/DefaultLPA /LOBControlblock Pool/PersistenceManager/PersistentSpace/DefaultLPA /LOBPage	Disk LOB caching	While disk LOB pages (SAP generic allocator Pool/PersistenceManager/P /Page with SAP HANA <= 2 LOB allocators Pool/PersistenceManager/P /LOBPage (data) and Pool/PersistenceManager/P /LOBControlblock (metadat Populating and releasing th rules like described for Pool/PersistenceManager/P /Page. You can influence th LOB caching behavior as de ("How can the SAP HANA r disk LOBs?"). Starting with determine the tables being SQL: "HANA_LOBs_LOBFi column MEM_SIZE_MB) re M_TABLE_LOB_STATISTI byte).
Pool/PersistenceManager/PersistentSpace(0)/Default LPA/Page Pool/PersistenceManager/PersistentSpace/DefaultLPA /Page Pool/PersistenceManager/PersistentSpace/DefaultLPA /DataPage	SAP HANA page cache	The SAP HANA page cache similar to a file system cach access to disk LOBs (SAP I HANA 2.0 SPS 03 the alloc ".../DataPage" and at the sa the dedicated allocator ".../L content of the allocator in te "pageaccess a" with hdbcon SQL: "HANA_Memory_Pag SAP HANA >= 2.0 SPS 01) a high amount of LOBs, you described in SAP Note 2220 memory utilization be influe garbage collection doesn't t 2169283), LOB related pag there can be an increased g allocator. The page allocato recovery, e.g. during a near

		HANA system replication or See SAP Note 2427897 for 1.00.122.06 - 1.00.122.16, 2.00.030 it is recommended (global.ini -> [persistence] - SAP Note 2600030) in orde page cache. The pages cac garbage collection issues re history files can implicitly re allocation stack modules ca DataAccess::GarbageColle 2169283 for more informatio collection. Delta merges (S the data of the new main st during delta merges of large significantly grow. The alloc delta merge is finished. You tables (SAP Note 2044468) space overhead. If the page weight statistics server cons you use check action check whenever possible (SAP No page cache also contains th Note 1871386). If you use p the following details: Check SAP Note 2111649 and co contributes to the page cac SAP Note 2497016 it can h pages are pinned in memor paged attributes cache than 1.00.122.09, <= 2.00.002.0 also used to store pages du writing down the pages to d the pages from the backup, increase and the DataAcces step at the end of the resto pages are written to disk (S performance of backup, res Due to a bug the size of thi with Revisions 1.00.110 to automatic reclaims with imp consider the following proac 2301382): Rev. 1.00.110 - scheduling Rev. 1.00.122.0 unload_upper_bound config apply and you see unloads allocator is still large, it is lik not able to keep up with the should check your I/O stack to secondary system replica "How can pending savepoin secondary site be explained more information.

Pool/PersistenceManager/PersistentSpace(0)/DefaultLPA/ShadowPage Pool/PersistenceManager/PersistentSpace/DefaultLPA/ShadowPage	I/O flush shadow pages	In some scenarios a copy o the flush thread can write it phase Encrypted page Row deallocated as soon as the and acknowledged. A large high I/O write volume or tha with the flush thread activiti more information regarding
Pool/PersistenceManager/PersistentSpace(0)/PageChunk Pool/PersistenceManager/PersistentSpace/PageChunk	Clustering of small pages for write	This allocator is used to cor chunks before writing them the allocated space is relea size is usually a consequen handle the I/O requests fast bottlenecks in the I/O stack requests (e.g. due to big tab IMPORT operations). See S about I/O analysis in SAP H
Pool/PersistenceManager/PersistentSpace(0)/RowStoreLPA Pool/PersistenceManager/PersistentSpace/RowStoreLPA	Row store control blocks	This allocator contains row significantly in case of a larg 2222277 and make sure to size, e.g. via cleanup (SAP (SAP Note 1813245).
Pool/PersistenceManager/PersistentSpace(0)/RowStoreLPA/RowStoreSegment Pool/PersistenceManager/PersistentSpace/RowStoreLPA/RowStoreSegment	Row store data	This allocator contains row the secondary site of a syst used for caching of row stor be created efficiently during row store size on the primar following SAP HANA param uses heap memory instead > [row_engine] -> use_shar this is no recommended set in shared memory unless th change.
Pool/PersistenceManager/PersistentSpace(0)/RowStoreLPA/Superblock Pool/PersistenceManager/PersistentSpace/RowStoreLPA/Superblock	Row store superblocks	This allocator is used during row store superblocks (inclu populating the row store sha is typically 0, but on second remain with a large size for disposition). See SAP Note the row store at a reasonab 2388483) or defragmentatio size of the allocator causes you can temporarily disable the following parameter (SA [persistence] -> optimized_
Pool/PersistenceManager/PersistentSpace/StaticLPA/DataPage Pool/PersistenceManager/PersistentSpace(0)/StaticLPA/Page Pool/PersistenceManager/PersistentSpace/StaticLPA/Page	liveCache pages	This area contains the page operated as part of SAP HA aren't swappable. Starting w space is reclaimed automat memory is required, so a la Note 2593571 for more info

Pool/PersistenceManager/PersistentSpace/TempLPA/DataPage Pool/PersistenceManager/PersistentSpace/TempLPA/Page	Temporary pages	This heap allocator stores to on disk when SAP HANA is data of temporary no loggin "HANA_Tables_Temporary 'YES', ORDER_BY = 'SIZE' to check for large temporary grow during repartitioning a also be used in context of in replication (SAP Note 1999 (module DataAccess::Backu BackupMetaDataToDelete can use the 'pageaccess a' 2222218) or using SQL: "HA Note 1969700, SAP HANA determine the main compon Example: ### TemporaryPa Disposition hasRefs | Count Temporary yes | 1 262144 T yes | 102 6684672 TableCo yes | 1 65536 BackupMetaD 1375 23068672000 In this s 20 GB of the allocator size i delta_datashipping allocatio 2.00.040 BackupMetaDataT backup / shipping and so th evicted by a resource conta restarted. You can switch to logreplay in order to avoid t delta_datashipping.
Pool/PersistenceManager/TemporaryUnifiedTableContainer	Temporary L2 delta and paged attribute information	This allocator can grow in c created with NO LOGGING SQL: "HANA_Tables_Temp (TEMPORARY_TABLE_TY IS_COLUMN_TABLE = 'TR 1969700 and check from ap or cleanup of these tempora
Pool/PersistenceManager/UndoDirectory	Undo and cleanup file directory	This allocator contains undo can grow significantly if pers blocked. See SAP Note 216 to garbage collection and ta garbage collection issues.
Pool/PersistenceManager/UnifiedTable container Pool/PersistenceManager/UnifiedTableContainer	L2 delta and paged attribute information	This allocator contains pers new delta mechanism used delta) and paged attributes HANA 1.0 SPS 08 delta log The actual delta area in colu this allocator. See SAP Not merges are properly configu storage size of the tables re allocator can grow in cases collection is blocked (SAP N

Pool/PersistenceManager/UnifiedTableContainer/MVCC	Column store MVCC information	This allocator is used to kee by storing creation and dele Its size can be significant in table footprint. You can dete "HANA_GarbageCollection_ column MVCC_TOTAL_MB
Pool/PersistenceManager/VarSizeEntryFreeSpaceInformation	Free space handling of persistence containers with variable size	This allocator keeps track o containers that can contain containers are typically use 2220627) and the container large allocator size is free s having deleted a significant SAP HANA may reduce the because some free space is See SAP Note 2220627 ("C for more information about r LOBs.
Pool/PersistenceManager/VirtualFile entry TID map	Transient mapping for VirtualFile overwrite optimization	A large size of this allocator of existing disk LOBs. See S can reduce the number of d management and archiving (SAP HANA >= 2.0). Startin allocator is no longer requir other contexts using virtual Note 2400022) can result in in some cases.
Pool/PlanningEngine/Buffer	Planning engine buffer	This heap allocator is linked used by BW integrated plan housekeeping and consiste linked to the related plannin significant for a longer time, longer required planning se allocations (SAP Note 2169 be triggered manually?" -> " collection").
Pool/PlanningEngine/Compile	Planning engine compilation structures	If the allocator size is large activities, you can check if d planning sessions can help 2169283 -> "How can garba manually?" -> "Planning eng
Pool/PlanningEngine/Fox	Dictionaries for FOX formula executions	FOX formula executions by temporary helper structures Pool/PlanningEngine/Fox. T planning function is finished often in combination with Po loop in the FOX script that h can check if dropping no lon help to reduce the allocation garbage collection be trigge garbage collection").
Pool/PlanningEngine/LookupDict	Master data lookup	You can use SQL: "HANA_

	dictionary of planning engine	(OBJECT_TYPE = 'LOOKU Note 1969700 in order to ch the creation times. After a re empty and re-populated on "HANA_Memory_PlanningE 1969700) in order to drop n Starting with SPS 10 SAP H the cleanup. If these steps d no longer required planning allocations (SAP Note 2169 be triggered manually?" -> " collection").
Pool/planviz/column store/plans Pool/planviz/column store/plans/ParentCycleDetector Pool/planviz/column store/PlanVizContext Pool/planviz/column store/PlanVizContext/JsonAllocator Pool/planviz/common/final results Pool/planviz/common/strings Pool/planviz/sql layer/PlanVizContext Pool/planviz/sql layer/PlanVizContext/PlanVizParams	PlanViz details	These allocators are used b SAP Note 2119087). In orde reasonable level you should restricted as possible (in ter SAP Note Impacted Revisio 1.00.120 - 1.00.122.02 Due 10 to SPS 12 it can happen grow even after you have d have to restart in order to cl growth. <= 1.00.122.06 A si allocators after tracing) still with Rev. 1.00.122.07. Even these allocators can grow a deactivated, so in productio that plan trace isn't used.
Pool/QueryLanguage	Enterprise search processing	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al are processed in context of Try to avoid complex search freestyle search terms) on l
Pool/QueryMediator	Processing of complex filters	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. Relate mediator related modules in QueryMediator::FilterProces ion QueryMediator::FilterTra with SAP HANA SPS 10 an the problem is fixed in many check if specifying the hint I helps to reduce the memory 2142945 for more informatio
Pool/ReplicationLogReceiverAllocator	Table replication log receive buffer	This buffer is used for recei table replication (SAP Note log is replayed. Large sizes temporary high amount of re during replay.

Pool/ResourceContainer Pool/ResourceContainer/ResourceHeader	Metadata for memory objects	A large size of these allocat number of memory objects. "HANA_Memory_MemoryC directly query M_MEMORY number of existing memory objects for Persistency/Pag Persistency/Container/Virtu high number of cached disk case you can consider to ac 2403124), perform a cleanu based on SAP Note 238848 SAP HANA 2.0 to packed L Pool/ResourceContainer/Re headers that are never dest over time until SAP HANA i
Pool/ResultCache(for cached view)	Static result cache information	This allocator stores static r with SAP HANA SPS 12 an available as of SAP HANA S more information related to cache.
Pool/RowEngine/Communication	TCP/IP communication channel management	See SAP Note 2222200 and connections and inter-node order to reduce the size of t
Pool/RowEngine/CpbTree Pool/RowStoreTables/CpbTree	Row store indexes	Check via SQL: "HANA_Ro 1969700) if the size of the h of the row store indexes. If i memory leak exists that can SAP HANA. Upgrade to at l eliminate known memory le 1.00.85.03 and Rev. 1.00.9 collection is not necessarily allocator can unnecessarily Rev. 1.00.95 a fix is delivere more information related to size is in line with the index tables with indexes in row s SAP Note 2388483) or mov "HANA_Indexes_Overview" 1969700) if there are large that are not required and ca
Pool/RowEngine/GlobalHeap	Global, unspecific row engine data areas	This allocator is an unspecif As all significant memory al dedicated allocators, Pool/R allocate too much memory. SAP Note Impacted Revisio a very high number of rows result in an increased alloca memory leak can be respon 2815963 2.00.040 - 2.00.04 activities If you face anothe allocation in Pool/RowEngin

		SAP incident for clarification
Pool/RowEngine/IndexRebuild	Row store index rebuild structures	This heap allocator is used typically during / after SAP I 2177064).
Pool/RowEngine/LOB	LOB data processed by database requests	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (default: <= 1024 byte) that requests. Larger LOB value Pool/RowEngine/QueryExe allocator is large, you shoul requesting a large amount c
Pool/RowEngine/LockTable Pool/RowStoreTables/LockTable	Row store lock and version information	A large size of this allocator transactional locks (SAP No issues (SAP Note 2169283) following known SAP HANA of this allocator: SAP Note I 2391552 1.00.110 - 1.00.12 Pool/RowStoreTables/Lock
Pool/RowEngine/MonitorView Pool/RowEngine/MonitorView/<monitor_view_name> Pool/RowEngine/MonitorView/M_MERGED_TRACES Pool/RowEngine/MonitorView/StatisticsMonitors/<monitor_view_name> Pool/RowEngine/MonitorView/StatisticsMonitors/M_CONTEXT_MEMORY Pool/RowEngine/MonitorView/StatisticsMonitors/M_DEV_INDEX_COLUMNS Pool/RowEngine/MonitorView/StatisticsMonitors/M_UNDO_CLEANUP_FILES	Monitoring view information	These heap allocators conta monitoring views (M_* view describes how the amount c controlled. In cases where t isn't part of the allocator nar areas within Pool/RowEngir hdbcons (SAP Note 222221 general you have to make s critical monitoring views (e.g that more selective queries for M_MERGED_TRACES Monitor views Impacted rev M_CS_ALL_COLUMNS M_ M_FUZZY_SEARCH_INDE >= 1.00.110 1.00.120 - 1.00 monitoring views can result conditions like scale-out and fulfilled. M_CS_LOADS M_ 2340582 The load and unlo memory leak. M_EXPENSIV 2112732 If most space is al "ptime::ExpensiveStatemen ringBuffer" at ExpensiveSta caused by the expensive st allocation by increasing the expensive statements trace HANA 1.0 SPS 07 and SPS be eliminated by deactivatin following parameter: global. use_in_memory_tracing = f 2.00.040 A memory leak is

		[expensive_statement] -> u false. Using the default of 't M_MERGED_TRACES 238 trace information that can b M_MERGED_TRACES wit "HANA_TraceFiles_Conten reasonable restriction (e.g. END_TIME). M_SQL_PLAN_CACHE_FO < 1.00.100 2186299 The SC embedded statistics server demand. M_TABLE_LOB_F significant amount of this m ptime::TableLobFilesMonito LOBs exist. Starting with SA consider using M_TABLE_L variant for M_TABLE_LOB_ memory requirements.
Pool/RowEngine/QueryCompilation	Compilation memory	This allocator is required du Large sizes can be caused Although parsing related thi statement memory limit. Yo SAP HANA issues resulting SAP Note Details 2124112 statements it can be require parameter: indexserver.ini <segment_size_in_byte> A Pool/RowEngine/QueryCom 2453348 The size of this al activated plan trace. A mem module AnalyticalAuthoriza AsQoStructure is a consequ with Rev. 1.00.122.09. 252 analytic privilege checks ca growth of the allocator. This 1.00.112.07, >= 1.00.122.0 2124112 Reducing the com ("Which problems and solut "High sampling overhead") size of Pool/RowEngine/Qu queries with long IN lists ar 2570371), a large allocator 2.00.024.02 and <= 2.00.03 context of many IFNULL fu size. As a workaround you unfolding (e.g. using the NO This issue is fixed with SAP SAP HANA <= 1.00.122.23 context of query rewriting c module ptime::qo_Normaliz sizes of this heap allocator. <= 2.00.037.00 and 2.00.04

		is possible in context of recu ptime::qo_Normalizer::trans mal_form calls. The NO_PA as a workaround. 2119087 an increasing allocator size trace only if really required a
Pool/RowEngine/QueryCompilation/SqlOptAlloc/qoContextAlloc	SQL optimization details	This heap allocator holds de The size is usually reasona number of allocator instantia M0470 ("Heap allocators wi HANA Mini Checks (SAP N potentially critical. Usually th ignored. Starting with SAP H will be shared and the high disappears.
Pool/RowEngine/QueryExecution Pool/RowEngine/QueryExecution/SearchAlloc	Row engine results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. To a m context allocates memory in also be a consequence of a connection contexts. To a c per connection / statement "HANA_Memory_ContextM '%Pool/RowEngine/QueryE SAP Note 1969700. Be awa allocations via generic impli the explicit context name. A following known SAP HANA of this allocator: SAP Note I 2000792 1.00.67 - 1.00.69.0 parallelized sub plan 22712 1.00.110 Batch INSERTs in JDBC >= 2.3.37 The transa more and more connection the allocator sizes over time
Pool/RowEngine/RowTableManager/MVCCManager/MVCCAllocator	Row store MVCC version management	This heap allocator is used during recovery operations replication sites and during starting with SAP HANA 2.0 03 it can increase in case o Note 2169283). The followin result in an increased alloca Revisions Details 2573738 store garbage collection on (SAP Note 1999880) with o 2.00.044 The rollback of an any DDL statements or DM causes a memory leak.
Pool/RowEngine/RSTempPage	Temporary row store tables	This allocator holds data rel store (SAP Note 2800007).

		temporary row store tables that sessions are closed wh drop related temporary tabl "HANA_Tables_Temporary 'NO', ORDER_BY = 'SIZE') identify the largest existing SAP Note 2000003 ("What tables can be created with S related to temporary and NO can check the following kno increased sizes of this alloc Details 2368929 <= 1.00.11 tables without variable leng 1.00.120 - 1.00.122.02 Mem tables are dropped
Pool/RowEngine/Session	Session management	Check if there is an unusua and eliminate the root caus
Pool/RowEngine/SQLPlan Pool/RowEngine/SQLPlanInfos	SQL cache	The SQL cache can be con underlying issues like a lack LIST sizes are not recogniz make sure that the SQL cac required. Be aware that the three times larger than the s following SAP HANA param plan_cache_size Reason: In itself, this allocator includes allocations such as data str cache, monitoring view data Note 2502256). Due to a bu <= 2.00.012.03 and <= 2.00 statistics weren't considered and so the heap allocator c to a bug on SAP HANA <= allocations were erroneousl SQL cache size (SAP Note
Pool/RowEngine/TableDMLRuntimeData	Table statistics	This allocator stores inform operations per table that ca M_TABLE_STATISTICS or "HANA_Tables_TablesStati increased size is usually ca tables. Reducing the numbe reduce the allocator size. S limit the amount of data in M reducing the allocator size. collection should usually no
Pool/RowEngine/TableRuntimeData	Table runtime data	This allocator contains table timestamp and record coun caused by an unusual high number of tables in the syst size. With SAP HANA Rev. happens when a temporary

		as of Rev. 1.00.97.02.
Pool/RowEngine/Version Pool/RowStoreTables/Version	Row store version space	A high number of versions r consistency (MVCC) reasor transaction. This increases "Which options exist to redu issues?" -> "Transactional p detailed recommendations.
Pool/RowEngine/ViewCache	Static result cache information	This allocator stores static r HANA SPS 11. The static re HANA SPS 11. See SAP N related to the SAP HANA st
Pool/RowTableUpdateAllocator	Row table update information	This allocator is used in cor Due to a SAP HANA bug it released in time. As a worka cache: ALTER SYSTEM CL is fixed with SAP HANA >=
Pool/SearchAPI Pool/SearchAPI/Itab Search	Intermediate results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al with SAP HANA Rev. 1.00.1 Pool/itab, so you can check Among others it is used by t Pool/hierarchyBlob for more
Pool/Search/PreparedQuery	Prepared searches	This allocator is used for se statements. The life time of SQL cache entry, so a large responsible for an increased cache would implicitly also c Note 2124112 for more info parsing and the SQL cache
Pool/SerializedObject	Fulltext index data structures	You can run SQL: "HANA_I = 'FULLTEXT', ORDER_BY 1969700 to display the exis 2800008) sorted by size. Cl indexes are really required. REPOSRC~SRC may exist Search (SAP Note 1918229 transaction SFW5.
Pool/SingleValueCacheBuilder	Single value cache	This heap allocator is used Note 2502256).
Pool/spatialcs	Spatial data	This allocator is linked to th 2091935) and it is typically clustering and for providing memory leak can be respon SAP HANA Revisions betw is available with Revision 1.
Pool/SQLChecker	Rule based semantic check during parsing	This allocator is used for a r SQL parsing (SAP Note 212

		to allocator Pool/SQLParse can check there for possible
Pool/SQLParserGlobal/SQLParserParse	Parsing	This allocator is used in cor 2124112). The following iss with SAP HANA >= 2.00.03 leak fixed with SAP HANA >
Pool/SQLPreprocessor	Preprocessing / rewriting after parsing	This allocator is used for pr parsing (SAP Note 212411 allocator Pool/SQLParserG check there for possible roo
Pool/SQLScript/Execution	SQLScript runtime information	Check for design problems If you face a high memory c 1.00.101, a bug can be res context of XS engine calls. in order to fix it.
Pool/SQLScript/Execution/Code	SQLScript runtime information	This allocator holds runtime procedures. The life time of cache entry is evicted, not v ends. So in case of an incre is an unusual high amount c procedure calls (SAP Note with SAP HANA <= 2.00.04
Pool/SQLScript/Execution/ManagedInvoke	SQLScript plan executions in L	This allocator is used for m SQLScript plan executions VARCHAR, NVARCHAR o e.g.: declare var lob; while . memory isn't released until memory requirements can b iterations. Check if you can avoid this critical scenario.
Pool/Statistics	Internal statistical information	See "How can allocations i optimized?" below for detai
Pool/StatisticsServer/ThreadManager/Stats::Thread_<num> Pool/StatisticsServer/JobManager/Stats::Thread_<num> Pool/StatisticsServer/JobManager/WriteLastValuesJob Pool/StatisticsServer/LastValuesHolder	Standalone statistics server	These allocators can becon statistics server is used and data is available (e.g. large connections). In order to op proceed as described at "W of SAP HANA memory issu optimizations" above.
Pool/StringContainer	Storage of (uncompressed) strings during column store activities	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. A tem Pool/StringContainer is pos amounts of data, e.g.: Data SAP HANA >= 1.00.122.11 the utilization reduces agai normally not critical.
Pool/TableConsistencyCheck	Table consistency	This allocator is related to t

	check	CHECK_TABLE_CONSIST You can limit the number of tables or run it at times with to reduce the risk of critical
Pool/Text/AEText Pool/Text/AEText/phrase_index Pool/Text/AEText/split_document_index Pool/Text/AEText/split_positional_index Pool/Text/AEText/termset_container Pool/Text/AEText/text_property_index	Fulltext index data structures	These allocators store spec indexes (SAP Note 280000 the size of existing fulltext in SAP HANA indexes in gene
Pool/TransientMetadataAlloc	Transient metadata	This allocator stores tempor definitions; local on transac life time of some data is link check if this cache is define Note 2124112). The followin allocator exist: SAP HANA 112.01 can suffer from a me called (SAP Note 2312994)
Pool/TransMgr	Transaction management	This pool contains data requ The life time of data is limite large size can be related to transactions. You can use S "HANA_Transactions_Curre 1969700) to display all exist number of transactional lock responsible for allocator gro "HANA_Locks_Transaction check for currently existing size can also increase when (SAP Note 2169283). With the record lock structures m so the allocator can grow un SAP HANA >= 2.00.034.
Pool/CacheFramwork/CacheMgr/CalcEngineNodeCache Pool/TREXCache/CacheMgr/CalcEngineNodeCache	Calculation engine node cache	This allocator holds informa engine node cache. See SA information.
Pool/CacheFramework/CacheMgr/CE_ScenarioModelCache Pool/TREXCache/CacheMgr/CE_ScenarioModelCache Pool/CacheMgr/CE_ScenarioModelCache	Calculation engine model cache	This heap allocator is used cache. Its size can be contr parameter (unit: KB): indexs max_cache_size_kb Startin cache can be cleared using SYSTEM CLEAR CACHE (SAP Note 2502256 for more
Pool/CacheFramework/CacheMgr/cs_access_statistics Pool/TREXCache/CacheMgr/cs_access_statistics	Access Statistics Cache	This allocator contains colu information in case the acce SAP Note 2785533 for more use it. Starting with SAP HA cleared using the following CLEAR CACHE ('cs_acces
Pool/CacheFramework/CacheMgr/CS_QueryResultCache[Realtime]	Query result cache	These heap allocators are l (SAP Note 2014148). Starti

Pool/CacheFramework/CacheMgr/CS_QueryResultCache[TimeControlled] Pool/TREXCache/CacheMgr/CS_QueryResultCache[Realtime] Pool/TREXCache/CacheMgr/CS_QueryResultCache[TimeControlled] Pool/CacheMgr/CS_QueryResultCache[Realtime] Pool/CacheMgr/CS_QueryResultCache[TimeControlled]		cache can be cleared using SYSTEM CLEAR CACHE (ALTER SYSTEM CLEAR C [TimeControlled]')
Pool/CacheFramework/CacheMgr/CS_StatisticsCache Pool/TREXCache/CacheMgr/CS_StatisticsCache Pool/CacheMgr/CS_StatisticsCache	Column store statistics cache	This heap allocator is used (SAP Note 2502256). Starti cache can be cleared using SYSTEM CLEAR CACHE (
Pool/CacheFramework/CacheMgr/Currency/UnitConversion_RateQueriesResultCache Pool/TREXCache/CacheMgr/Currency/UnitConversion_RateQueriesResultCache Pool/CacheMgr/Currency/UnitConversion_RateQueriesResultCache	Currency conversion cache	This heap allocator is used (SAP Note 2502256). It is a parameter settings: indexse global indexserver.ini -> [bu cache_erp_currency_query > [businessdb] -> cache_er cache is invalidated when u tables like TCURR are mod SPS 03 the cache can be c command: ALTER SYSTEM ('UnitConversion_RateQuer
Pool/CacheFramework/CacheMgr/DataStatisticsAdviserCache Pool/TREXCache/CacheMgr/DataStatisticsAdviserCache Pool/CacheMgr/DataStatisticsAdviserCache	Data statistics adviser cache	This heap allocator is used (SAP Note 2502256). Starti cache can be cleared using SYSTEM CLEAR CACHE (
Pool/trex_wrapper_body	P*TIME / TREX wrapper	This allocator is used to wra e.g. prepared execution pla engine functionalities. Its life in the SQL cache - once the memory in Pool/trex_wrapp growth can be linked to a hi scenarios, e.g. reported by calculation scenarios") repo (SAP Note 1999993). Make scenarios remains on a rea 2124112 for more informati cache.
Pool/AFL_UDF_CORE	Intermediate UDF AFL results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al Demand Forecast (UDF) AF Customer Activity Reposito is released once the functio
Pool/AFL_XRP	Intermediate XRP AFL results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (eXtended RePlenishment) Customer Activity Reposito

		is released once the functio
Pool/OSA/AnalyticalToolOSA	Intermediate results of the AnalyticalToolOSA step of the POS AFL	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (Point Of Sales On Shelf Av SAP Customer Activity Rep memory is released once th
Pool/OSA/CreateIntraWeekPattern	Intermediate results of the CreateIntraWeekPattern step of the POS AFL	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (Point Of Sales On Shelf Av SAP Customer Activity Rep memory is released once th
Pool/OSA/EstimateModel	Intermediate results of the EstimateModel step of the POS AFL	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (Point Of Sales On Shelf Av SAP Customer Activity Rep memory is released once th
Pool/OSA/MonitorOSA	Intermediate results of the MonitorOSA step of the POS AFL	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (Point Of Sales On Shelf Av SAP Customer Activity Rep memory is released once th
Pool/OSA/Singleton	Intermediate results of all steps of the POS AFL	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al (Point Of Sales On Shelf Av SAP Customer Activity Rep memory is released once th
Pool/UdivListMgr/UdivListContainer	MVCC management	This allocator is responsible concurrency control (MVCC transactions. In order to che transactions you can proce exist to optimize the SAP H "Transactional problems".
Pool/ValueArray Pool/ValueArrayColumnDeserialize	Join engine results	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. These Pool/JoinEvaluator/JERequ Pool/ValueArray was a prev with current Revisions. Poo used when join engine resu

		another in scale-out scenar
Pool/XDictData	Intermediate OLAP engine dictionary data	See question "Which gener the SQL statement memory make sure that SQL statem efficient as possible. This al query processing in the OL/ if columns with many distinc don't exist or aren't pushed
Pool/XSEngine/XSJobScheduler/_<application> Pool/XSEngine/XSJobScheduler/_<application>/_<job_name>	XS engine jobs	These heap allocators conta jobs. Every job creates an c the number of allocator inst The size is at the same time Starting with SAP HANA 1.(2.00.024 jobs with the same so the amount of allocator ir lower. There is no way to pu After a SAP HANA restart tl scratch.
StackAllocator	Thread stacks	This allocator contains threa typically caused by a large r settings of the following stac [threads] -> default_stack_s worker_stack_size_kb Thes remain on default values, ac specific situations (e.g. SAF
VirtualAlloc	SAP HANA external memory management	This allocator is related to m standard SAP HANA memo Java Virtual Machine (JVM)

14. How can I identify how a particular heap allocator is populated?

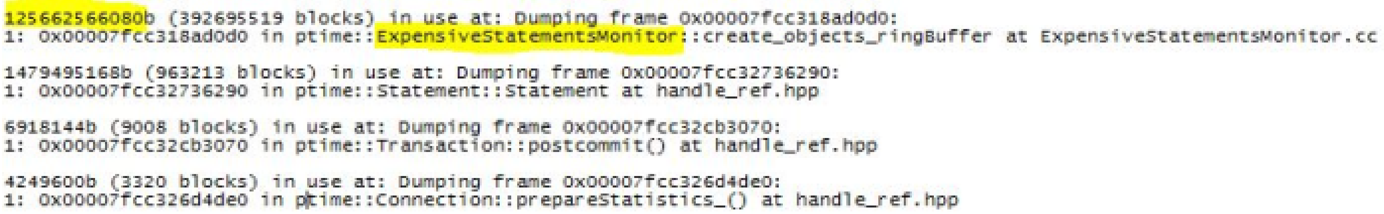
You can use the tool hdbcons on operating system level in order to understand better how a heap allocator is filled (SAP Note 2222218). Typical commands are:

Command	Example	Purpose
help mm		Overview of all memory management (mm) related command options
mm bl -t <allocator>	mm bl -t Pool/Statistics	Show top memory contributors ("block list") in <allocator> sorted by used size descending
mm cg -o <file>.dot <allocator>	mm cg -o callgraph.dot Pool/Statistics	Generate output file with allocation stack trace information for <allocator>
mm f <allocator> as	mm f Pool/Statistics as	Activation of allocator call stack trace for <allocator> Particularly useful in case of suspected memory leaks so that you can understand from which modules the memory allocations are mainly performed Can result in significant overhead and should only be activated for limited times
mm f <allocator> as -d	mm f Pool/Statistics as -d	Deactivation of allocator call stack trace for <allocator>
mm ru	mm ru	Reset all previous measurements ("reset usage")

mm top -l <num> <allocator>	mm top -l 20 Pool/Statistics	Generate report with top <num> call stacks recorded for <allocator>
pageaccess a		Provide breakdown of Pool/PersistenceManager/PersistentSpace/DefaultLPA/* allocators content based on page type, e.g.: ConvIdxPage 256k Temp : 1 (262144 Byte) ConvLeafPage 256k Temp : 130 (34078720 Byte) TidCidMappingPage 256k Short : 1 (262144 Byte) FileIDMappingPage 256k Temp : 172 (45088768 Byte) FileIDMappingPage 256k Short : 2 (524288 Byte) ContainerDirectoryPage 256k Short : 35 (9175040 Byte) ContainerDirectoryPage 256k Long : 2 (524288 Byte) ContainerNameDirectoryPage 256k Long : 1 (262144 Byte) UndoFilePage 64k Short : 707 (46333952 Byte) VirtualFilePage 4k InternalShort : 134 (548864 Byte) VirtualFilePage 16k InternalShort : 57 (933888 Byte) VirtualFilePage 64k InternalShort : 325 (21299200 Byte) VirtualFilePage 256k InternalShort : 196 (51380224 Byte) VirtualFilePage 1M InternalShort : 552 (578813952 Byte) VirtualFilePage 4M InternalShort : 2832 (11878268928 Byte) VirtualFilePage 16M InternalShort : 9458 (158678908928 Byte) VarSizeEntryBasePage 256k Short : 809 (212074496 Byte) ...

Example 1 (check for top memory contributors in allocator):

```
mm bl -t Pool/RowEngine/MonitorView
```



```
125662566080b (392695519 blocks) in use at: Dumping frame 0x00007fcc318ad0d0:
1: 0x00007fcc318ad0d0 in ptime::ExpensiveStatementsMonitor::create_objects_ringBuffer at ExpensiveStatementsMonitor.cc

1479495168b (963213 blocks) in use at: Dumping frame 0x00007fcc32736290:
1: 0x00007fcc32736290 in ptime::Statement::Statement at handle_ref.hpp

6918144b (9008 blocks) in use at: Dumping frame 0x00007fcc32cb3070:
1: 0x00007fcc32cb3070 in ptime::Transaction::postcommit() at handle_ref.hpp

4249600b (3320 blocks) in use at: Dumping frame 0x00007fcc326d4de0:
1: 0x00007fcc326d4de0 in ptime::Connection::prepareStatistics_() at handle_ref.hpp
```

This output indicates that more than 100 GB of allocator Pool/RowEngine/MonitorView is consumed by the ExpensiveStatementsMonitor and so optimizations like adjustments to the expensive statements trace or implementing a bugfix to resolve a memory leak problem can be considered.

Example 2 (create an allocator call stack trace and extract top 5 call stacks)

```
mm ru mm f Pool/Statistics as -- Now wait until a representative amount of allocations is
captured mm top -l 5 Pool/Statistics mm ru mm f Pool/Statistics as -d
```

15. How often are OOM dumps written?

In case of OOM situations SAP HANA may write a dump, e.g.:

- <service>_<host>.<port>.**rtedump**.<timestamp>.**oom**.trc
- <service>_<host>.<port>.**rtedump**.<timestamp>.**after_oom_cleanup**.trc
- <service>_<host>.<port>.**rtedump**.<timestamp>.**compositelimit_oom**.trc
- <service>_<host>.<port>.**rtedump**.<timestamp>.**oom_memory_release**.trc

For more details about the different dump types see SAP Note 2000003 ("Which types of dumps can be created in SAP HANA environments?").

Not every OOM termination results in an OOM dump because in case of a memory bottleneck many different transactions can run into an OOM error within a short time frame. Per default a SAP HANA service only creates an OOM dump if the last dump was written at least one day ago. This behaviour can sometimes be of disadvantage when two individual OOM situations should be analyzed that happened within less than 24 hours.

In special cases you can reduce the minimum time between two OOM dumps using the following SAP HANA parameter:

```
global.ini -> [memorymanager] -> oom_dump_time_delta = <min_seconds_between_oom_dumps>
```

If you set the parameter for example to 7200, the minimum interval between two OOM dumps will be two hours (7200 seconds).

16. Where can I find more information regarding SAP HANA memory consumption?

The document SAP HANA Memory Usage Explained provides a good overview of different types of memory in SAP HANA environments.

17. How can the resident memory be smaller than the allocated memory?

Normally the allocated memory should be fully contained in the resident memory, nevertheless there are a few exceptions:

- If parts of the virtual memory are paged out to disk, the resident memory can be smaller than the allocated memory.
- There are technical constellations where parts of the heap memory and the row store shared memory are marked as used, but not as resident.

18. What are typical reasons for significant size differences in memory vs. on disk?

The allocation of tables in memory and on disk may significantly differ for the following reasons:

Reason	Symptom	Details
No logging tables	Memory > disk	Tables created with the NO LOGGING option are not persisted to disk. See SAP Note 2000003 ("What kind of temporary and non-persisted tables can be created with SAP HANA?") for more information.
Temporary tables	Memory > disk	Tables created with the TEMPORARY option are not persisted to disk. See SAP Note 2000003 ("What kind of temporary and non-persisted tables can be created with SAP HANA?") for more information.
Single column and row store indexes	Memory > disk	Single column indexes and row store indexes aren't persisted to disk. See SAP Note 2160391 ("Are indexes persisted to disk?") for more information.
Logically moved tables	Memory > disk	If tables are moved logically, their disk footprint can be significantly smaller than the size in memory. See SAP Note 1994408 for more information.
Hybrid LOBs	Disk > memory	Large hybrid LOB values (SAP Note 1994962) are not loaded into memory, so the disk size of tables is larger than the memory size.
Partially loaded tables	Disk > memory	If columns of a table are only partially loaded into the memory, the disk size is larger than the current memory size. You can use SQL: "HANA_Tables_LargestTables" (SAP Note 1969700) to check disk size, potential maximum memory size and current memory size.
Paged attributes	Disk > memory	If columns are defined as paged attributes (SAP Note 1871386), e.g. in data aging contexts, columns with are not completely loaded into memory. Instead they are loaded into a (smaller) paged memory pool if required.
Data fragmentation	Disk > memory	A fragmented data area can significantly increase the disk requirements. You can use SQL: "HANA_Disks_Overview" (SAP Note 1969700) to check

		for the amount of fragmentation on disk side.
L2 delta migration	Disk > memory	When upgrading from SAP HANA <= SPS 08 to SAP HANA >= SPS 09 an L2 delta migration takes place that can temporarily increase the disk space requirements significantly. See SAP Note 2349081 for more information.
Large MVCC size	Disk > memory	MVCC information (SAP Note 2169283) can allocate additional space on disk. SAP Note 2146989 discusses a specific MVCC issue in context of upgrades to SAP HANA 1.0 SPS 09. You can use SQL: "HANA_Tables_DiskSize_UnifiedTabl es" (SAP Note 1969700) in order to check for table disk sizes including MVCC space.
Activities with heavy data movement (table redistribution, migration, data load, delta merge or optimize compression of large tables)	Disk > memory	Processing a larger amount of data can result in an temporary increase of disk space requirements for various reasons (shadow pages, snapshots, uncompressed columns, interim tables, ...). For that reason the Storage Whitepaper available via SAP Note 1900823 recommends to make sure that the double data size should be used during operations like table redistribution or migration import.
Database snapshots	Disk > memory	Database snapshots can result in significantly increased disk space requirements, because the before image of modified blocks needs to be stored in addition to the normal data blocks. Therefore you should make sure that old snapshots are deleted. SQL: "HANA_IO_Snapshots" (SAP Note 1969700) can be used to check for old snapshots. See SAP Note 2100009 for more information related to savepoints and snapshots. You can use the hdbcons command "snapshot a <snapshot_id>" (SAP Note 2222218) to find out how much disk space is allocated due to a snapshot. In the output you can find the relevant size information: dropping this snapshot would free <num_pages> pages with total size of <size_MB> MB See SAP Note 2100009 ("What are reasons for snapshots being retained for a long time?") for typical situations when snapshots exist for a long time.
Low savepoint frequency	Disk > memory	Normal savepoints work in a similar way like database snapshots and all pages referenced by one savepoint are kept until the next savepoint succeeds. If a page is changed, both the former version and the new version needs to be stored in parallel. Normally this doesn't result in a significantly increased disk space, but in case of a low savepoint frequency (e.g. due to a very long running savepoint or due to a high setting of parameter global.ini -> [persistence] -> savepoint_interval_s) or in case of a high change load the persistence overhead can be significant. See SAP Note 2100009 for more information related to savepoints and snapshots. In this scenario you can observe a rising amount of shadow pages and check ID 383 ("Max. size of shadow pages (GB, last day)") of the SAP HANA Mini Checks (SAP Note 1999993) can be reported as potentially critical.
Garbage collection blocked	Disk > memory	Blocked persistence garbage collection can result in a significant increase of disk space. SAP Note 2169283 describes how to analyze issues with garbage collection.
Large DELETE / TRUNCATE	Disk > memory	As described in SAP Note 2014987 the disk size can remain at a high level after having performed a large DELETE or TRUNCATE operation. The amount of allocated disk space can be 16 MB * <num_columns> * <num_partitions> in the worst case. Proceed as described in SAP Note

		2014987 in order to reduce the allocated disk size. Be aware that with SAP HANA <= 2.00.024.06 and <= 2.00.034 packed LOBs of column store tables are generally not purged after a TRUNCATE operation. You have to upgrade to a more recent SAP HANA Revision or recreate the table to purge the midsize LOBs successfully. See SAP Note 2707020 for more details.
Orphan disk LOBs	Disk > memory	Orphan disk LOBs can be responsible for a significant space allocation on disk that isn't reflected in the memory. See SAP Note 2220627 ("Can there be orphan disk LOBs?") for more information related to orphan disk LOBs.
Orphan persistence files	Disk > memory	SAP Note 2400005 -> "Can there be orphan files on persistence level?" describes scenarios of orphan pages left on persistence side and possible cleanup options. SAP Note 2910004 describes a SAP HANA bug in context of packed LOBs and DDL operations with SAP HANA <= 2.00.046 that can result in increased disk space requirements because no proper cleanup happens on persistence level.
Virtual files	Disk > memory	In some scenarios, e.g. in context of smart data integration (SDI, SAP Note 2400022), virtual files are used that aren't related to tables. If they are loaded into memory, they are part of the SAP HANA page cache (Pool/PersistenceManager/PersistentSpace/DefaultLPA/Page) that will be unloaded early. The size on disk is typically significantly larger than the memory footprint. You can use the following hdbcons command to determine the largest virtual files of a service (SAP Note 2222218): hdbcons 'vf t'
LOB fragmentation	Backup > disk	Although this kind of space overhead doesn't properly fit here (because disk LOBs are never loaded into memory), it should be mentioned for completeness purposes. LOBs are allocated with fix page sizes (>= 4 KB) and so there can be significant unused space, particularly if you have many small LOB values smaller than 4 KB. See SAP Note 2220627 for more information related to LOBs. You can use SQL: "HANA_LOBs_LOBFiles" (SAP Note 1969700) in order to check for allocated LOB space (PHYS_SIZE_MB) and used LOB space (BIN_SIZE_MB). Due to the fact that the full pages are backed up, the backup size can be significantly larger than the used disk size in some cases.

19. Which general optimizations exist for reducing the SQL statement memory requirements?

The following heap allocators are mainly used in context of processing database requests (e.g. for intermediate result sets and structures) and usually their life time ends when the database request is finished:

- Pool/AttributeEngine/Transient
- Pool/AttributeEngine/Transient/updateContainerConcat
- Pool/CSPlanExecutor/PlanExecution
- Pool/DocidValueArray
- Pool/ESX
- Pool/ExecutorPlanExecution
- Pool/Filter
- Pool/itab
- Pool/itab/VectorColumn
- Pool/JoinEvaluator
- Pool/JoinEvaluator/DictsAndDocs
- Pool/JoinEvaluator/JEAggregate

- Pool/JoinEvaluator/JEAggregate/Results
- Pool/JoinEvaluator/JEAssembleResults
- Pool/JoinEvaluator/JEAssembleResults/Results
- Pool/JoinEvaluator/JECalculate
- Pool/JoinEvaluator/JECalculate/TmpResults
- Pool/JoinEvaluator/JECalculate/Results
- Pool/JoinEvaluator/JECreateNTuple
- Pool/JoinEvaluator/JEEvalPrecond
- Pool/JoinEvaluator/JEPlanData/deserialized
- Pool/JoinEvaluator/JEPreAggregate
- Pool/JoinEvaluator/JERequestedAttributes/Results
- Pool/JoinEvaluator/JEStep1
- Pool/JoinEvaluator/JEStep2
- Pool/JoinEvaluator/NTuple
- Pool/JoinEvaluator/PlanDataAttrVals/Deserialized
- Pool/JoinEvaluator/ValueList
- Pool/L/llang/Runtime/Global
- Pool/L/llang/Runtime/Local
- Pool/malloc/libhdbcalcengine.so
- Pool/malloc/libhdbcalcengineapi.so
- Pool/malloc/libhdbcalcenginepops.so
- Pool/malloc/libhdbcs.so
- Pool/malloc/libhdbcsapi.so
- Pool/malloc/libhdbcswrapper.so
- Pool/malloc/libhdbevaluator.so
- Pool/malloc/libhdbitab.so
- Pool/malloc/libhdbolap.so
- Pool/mds
- Pool/mds/CubeAxis
- Pool/parallel/aggregates
- Pool/parallel/align
- Pool/parallel/compactcol
- Pool/parallel/ihm
- Pool/parallel/pop
- Pool/parallel/temp_aggregates
- Pool/parallel/temp_dimensions
- Pool/parallel/temp_other
- Pool/QueryLanguage
- Pool/RowEngine/LOB
- Pool/RowEngine/MonitorView
- Pool/RowEngine/QueryCompilation
- Pool/RowEngine/QueryExecution
- Pool/RowEngine/QueryExecution/SearchAlloc
- Pool/SearchAPI
- Pool/SearchAPI/Itab Search
- Pool/StringContainer
- Pool/ValueArray
- Pool/ValueArrayColumnDeserialize
- Pool/XDictData

To a certain extent this specific allocator class can also be identified in monitoring view M_HEAP_MEMORY with COMPONENT = 'Statement Execution & Intermediate Results', but the assignment to this class is not always 100 % precise.

The following general rules can help to reduce memory requirements of SQL statements during execution (or in some cases life time of the SQL cache entry):

Rule	Details
As few rows as possible	Use as many restrictions as possible so that the amount of fetched records is as small as possible.
As few columns as possible	Select as few columns as possible. Avoid "SELECT *" whenever possible.
Avoid UNION ALL, UNION, INTERSECT, EXCEPT	These operations can't be handled by the column engine and so optimizations like late materialization (SAP Note 1975448) are not possible. As a consequence the memory requirements can significantly increase. Therefore you should use alternative whenever possible (e.g. OR instead of UNION or UNION ALL).
BW: Configure safety belt	If BW queries read a large amount of data, check if it is possible to configure the query safety belt as described in SAP Note 1127156.
Homogeneous user for composite provider / stacked calculation view on top of scripted calculation view	If the user of a composite provider / stacked calculation view and of an inner scripted calculation view is different, predicate pushdown may be impacted and so a high memory consumption related to intermediate result set allocators is possible. Either make sure that the owner is identical or define the scripted calculation view in "Invoker" mode.
Disable inlining	Procedure (e.g. AMDP) executions are typically executed in a monolithic way with all individual database requests being inlined. This increases complexity and imposes a risk of wrong optimizer decisions. In these scenarios it can sometimes help to use the NO_INLINE hint (SAP Note 2142945) so that every database request is executed individually.
Use bind variables	Make sure that bind variables are used (SAP Note 2124112) in order to minimize the number of similar statements that have to be stored in the SQL cache. This can reduce the utilization of allocators like Pool/malloc/libhdbcsapi.so that among others store information that exists as long as the underlying statement is part of the SQL cache.

If the memory consumption of these allocators remains at levels that can hardly be explained by executions of database requests, you can consider the following technical SAP HANA root causes:

Scenario	Details
Memory leak	If you see a steady size increase, it can be caused by a memory leak, e.g.: SAP Note 2062555 (join operation in the subquery of an UPDATE statement, fixed with Rev. >= 1.00.83) SAP Note 2088349 (querying calculation views with currency conversion, fixed with Rev. >= 1.00.84) SAP Note 2789785 (memory leak when calling virtual procedure in SDI, fixed with Rev. >= 1.00.122.25) If you suspect a memory leak that is not documented, yet, open a SAP incident on component HAN-DB in order to request a more detailed analysis.
Cancellations	In some cases cancellations don't clean up all allocations properly, resulting in memory leaks. For example, heap allocators Pool/JoinEvaluator/JERequestedAttributes/Results, Pool/parallel/aggregates and Pool/parallel/compactcol can increase when database requests are cancelled in temp indexes in context of itabs aren't cleared properly with SAP HANA <= 2.00.046. In general you can try to minimize the amount of cancellations and rollbacks. Furthermore you can watch out for cancellations and map them to allocator size increases to understand if there is a correlation.

SAP HANA internal reference still open	Normally the statement specific heap allocators should be relased as soon as the database request ends. Due to SAP HANA bugs it can happen that the cleanup isn't performed if certain references (like temporary tables) still exist. You can check if you suffer from this scenario by clearing the SQL cache globally or some suspicious entries individually. See SAP Note 2124112 ("How can entries in the SQL cache be invalidated or reparsed manually?") for more information. Attention: Clearing the SQL cache results in additional parsing requirements and so temporary performance regressions are possible. The following already known scenarios exist: SAP Note 2312976 (DML operations, problem exists for Rev. 1.00.100 - 1.00.102.06 and 1.00.110 - 1.00.112.01) SAP Note 2312983 (memory leak in Pool/parallel/aggregates when querying on distributed environment with SAP HANA 1.00.100 - 1.00.102.06 and 1.00.110 - 1.00.112.02) SAP Note 2533352 (no proper cleanup after execution, fixed with SAP HANA >= 1.00.122.13, >= 2.00.012.02 and >= 2.00.021) SAP Note 2535110 (Memory Leak on Pool/parallel/compactcol and Pool/parallel/aggregates or Pool/itab with SAP HANA <= 1.00.122.12, <= 2.00.012.01 and 2.00.020) If the size of statement allocators reduces significantly after clearing the SQL cache, you can use this approach as a workaround and additionally open a SAP incident on component HAN-DB in order to request a fix for this behavior.
No cleanup in context of terminations	The following scenarios can be responsible for an incomplete cleanup in case of terminations: SAP HANA Revisions <= 1.00.122.12, <= 2.00.002.02, <= 2.00.012.01 and 2.00.020 can suffer from an increase in Pool/itab in context of INSERT abortions (SAP Note 2535110). With SAP HANA <= 1.00.122.15 the smart data access (SDA, SAP Note 2180119) related internal procedure SDA_SELECT_AS_ITAB_DEV isn't OOM safe and so memory can remain allocated after an out-of-memory situation. With SAP HANA <= 1.00.122.15, <= 2.00.012.04 and <= 2.00.024.00 an OOM termination can result in an incomplete cleanup of memory allocations of distributed queries (SAP Note 2612022). With SAP HANA 2.00.040 - 2.00.044 terminations in context of calculation view accesses can result in memory leaks because statement related memory isn't released properly (SAP Note 2866563).
BW temporary tables	In BW environments the high utilization can be linked to temporary tables (SAP Note 2800007). In this case you can run report RSDDTMPTAB_DELETE to drop these temporary objects in order to check if it has a positive impact on the Pool/itab size (SAP Note 2352541). Also other allocators like Pool/malloc/libhdbolap.so, Pool/parallel/aggregates and Pool/parallel/compactcol have already shown significant growth in this context, e.g. in case of large CRM segmentation tables (0BW:CRM:BI0_0M*). Be aware that running this report can result in terminations of currently running reports. In order to make sure that SAP HANA drops a temporary table after having reached the retention time you can set the following parameter (SAP HANA >= 2.0): indexserver.ini -> [metadata] -> mem_usage_for_active = 0
Planning engine	If the allocator size is large in context of planning engine activities, you can check if dropping no longer required planning sessions can help to reduce the allocations (SAP Note 2169283 -> "How can garbage collection be triggered manually?" -> "Planning engine garbage collection"). SAP Note 2583148 describes a problem with missing garbage collection in context of the TMA application.
MDX	If you execute MDX queries (e.g. using SAP HANA Studio), make sure that you explicitly close MDX requests (MDX CLOSE REQUEST <guid>) when you no longer need them. A COMMIT will not automatically close the requests. If you suspect orphan MDX queries (e.g. because MDX CLOSE REQUEST wasn't executed), you can check for MDX related temporary tables in M_TEMPORARY_TABLES (MDX_..._<guid>). By dropping these tables (DROP TABLE _SYS_BIC.MDX_..._<guid>) also the related internal tables should be dropped. Only drop these tables if you are sure that they are no longer required.
BPC queries with MDX	If BPC reports are executed on the system, the results may not be closed properly in context of ENABLE_HANA_MDX = 'X' (SAP Note 2108247).

Smart data access	If you use smart data access (SAP Note 2180119) with Rev. <= 1.00.85.02 or Rev. 1.00.90 - 1.00.91, a SAP HANA bug can be responsible for growing Pool/itab requirements. Upgrade to a more recent SAP HANA Revision in order to resolve the problem. See SAP Note 2242507 for more information.

20. How can the tables with the highest memory consumption be determined?

You can use *SQL: "HANA_Tables_LargestTables"* (SAP Note 1969700) in order to check for the largest tables in memory. The following ORDER_BY settings are possible:

ORDER_BY	Details
MAX_MEM	The tables (including indexes and LOBs) with the highest possible maximum memory consumption are shown. The maximum memory information is independent of the currently loaded columns and so it provides a general overview independent of the current load state.
CURRENT_MEM	The tables with the highest current memory consumption (including indexes and LOBs) are displayed.
TABLE_MEM	The tables with the highest current memory consumption (excluding indexes and LOBs) are displayed.
INDEX_MEM	The tables with the highest index memory consumption are displayed.

Be aware that there are situations where the maximum memory information (M_CS_TABLES.ESTIMATED_MAX_MEMORY_SIZE_IN_TOTAL) is not filled properly, particularly after DDL operations with SPS 08 and below. If you have doubts you can user ORDER_BY = 'TOTAL_DISK' to display the tables with the highest disk space consumption.

21. How much swap space should be configured for SAP HANA hosts?

It is recommended to configure a small swap space in order to avoid performance regressions at times of high memory utilization on operating system side. Instead it is usually better if activities are terminated with "out of memory" errors. This makes sure that the overall system is still usable and only certain requests are terminated. A good value for the swap space is 2 GB (see e.g. SAP Note 1944799 for SLES environments).

22. What is memory garbage collection?

Memory garbage collection and defragmentation is done in order to release no longer used memory. It is not required to perform this task manually as SAP HANA will automatically take care for this activity whenever required. In exceptional cases you can trigger / configure memory garbage collection manually:

Command / Setting	SAP Note	Details
hdbcons 'mm gc -f'	2222218	This command triggers an immediate garbage collection. Defragmentation will happen as much as possible. Attention: Executing this command has potentially critical side-effects like a temporary blockage of business operations, a reduction of address space or - in the long run - increased memory fragmentation. Therefore it must only be executed when advised by SAP support.
global.ini -> [memorymanager] -> gc_unused_memory_threshold_abs	2169283	These parameters trigger a garbage collection when both the absolute and relative value is exceeded. As

global.ini -> [memorymanager] -> gc_unused_memory_threshold_rel		soon as one of the configured limits is reached, memory garbage collection stops. Attention: Setting these parameters can result in frequently recurring memory defragmentation activities and related performance regressions. If at all, you should set these parameters only temporarily (e.g. for a few minutes) during a less critical time frame. Unsetting the parameters will not stop the initial defragmentation. Attention: Due to a bug these parameters don't take effect with SAP HANA 2.00.040 - 2.00.042.

Attention: Setting these parameters can cause significant performance issues, so they shouldn't be used unless explicitly requested by SAP support.

The following problems are possible when triggering manual memory garbage collection:

Risk	Details
OOM situations	Each memory garbage collection has an impact on the virtual address space utilization and so the risk of out-of-memory terminations because of address space limitations increases. See "Which indications exist that an OOM situation is triggered by the operating system?" for more information.
Performance regressions	At runtime of a memory garbage collection SAP HANA internal lock contention can result in reduced performance and increased resource consumption. In busy systems contention and spin locks on operating system side are possible when releasing memory back to the operating system. This scenario results in increased system CPU consumption and page faults. As a workaround the following parameter can be set in order to execute the defragmentation sequentially: indexserver.ini -> [memorymanager] -> disabled_parallel_tasks = poolgarbagecollection

23. Why do I get an OOM although the SAP HANA allocation limits aren't reached?

The following reasons can be responsible for OOM situations although neither the global nor the process specific allocation limits aren't reached:

Reason	Details
Operating system memory exhausted	Check if the available memory is exhausted on operating system side, e.g. because of external software allocating a lot of memory, large caches or another SAP HANA instance. Make sure that in the future there is always enough physical memory available to host the complete SAP HANA allocation limit. See "Which indications exist that an OOM situation is triggered by the operating system?" below for more details.
Small temporary process allocation limit	Based on the defined allocation limits SAP HANA and the current service memory allocations the temporary process allocation limit (TPAL) may be significantly smaller than the defined allocation limit. As a consequence OOMs are possible although the configured allocation limits aren't reached. SAP Note 2133638 describes a related startup issue that can happen as of Rev. 90.
Statement memory limit reached	OOM dumps with "compositelimit" in their names are no global memory shortages. Instead they are linked to a defined statement memory limit. See "Is it possible to limit the memory that can be allocated by a single SQL statement?" above for more details.

24. How can I involve SAP to perform a detailed memory check?

A detailed SAP HANA memory check and further general health checks and performance optimiaztions are performed as part of the SAP HANA Technical Performance Optimization Service (TPO).

25. Why is the allocated memory in some heap allocators very large?

The column EXCLUSIVE_ALLOCATED_SIZE in monitoring view M_HEAP_MEMORY (respectively HOST_HEAP_ALLOCATORS) contains the sum of all allocations in this heap allocator since the last startup. Normally also a lot of deallocations happen, so the EXCLUSIVE_ALLOCATED_SIZE can be much higher than the currently allocated size. For example, if over time 100 MB are allocated and deallocated 10 times, the actual allocated size is 0, but EXCLUSIVE_ALLOCATED_SIZE would show 1 GB (10 * 100 MB).

If the overall allocated memory is much higher than the overall used memory, the difference is usually free for reuse, so no longer heap allocator specific. Therefore the EXCLUSIVE_ALLOCATED_SIZE information can only be used to understand which heap allocators have the highest "throughput" in terms of memory allocations, but it is not helpful to understand the current memory situation.

26. Why does PlanViz show a high "Memory Allocated" figure?

If you observe a high "Memory Allocated" figure in PlanViz (SAP Note 2073964) that may significantly exceed the configured statement_memory_limit setting, this is typically caused by the same reason like discussed in the previous question: PlanViz summarizes the overall memory allocation irrespectively of intermittent deallocations. As a consequence the recorded allocated memory can be much higher than maximum memory allocation at a specific point in time.

See SAP Note 2302903 for more information.

27. Why does the delta storage allocate more memory with SAP HANA SPS >= 09?

With SAP HANA SPS 09 the delta storage was significantly adjusted. As a consequence the minimum memory footprint of the delta storage of a loaded empty column increased from around 2 KB to more than 8 KB. Having many empty tables with many columns this can increase the overall delta storage size by 10 GB and more. This is an expected behavior that can't be changed.

28. Are there any special memory considerations for multitenant databases?

In multitenant database container (MDC) scenarios (SAP Note 2101244) you should make sure that individual containers don't consume excessive amounts of memory, impacting other containers or the system database. On tenant level the memory can be controlled by the service specific parameter global.ini -> [memorymanager] -> allocationlimit in the best way. Optimally the sum of all tenant allocation limits sums up to the global allocation limit, but it is also possible to exceed it.

Example:

- Global allocation limit: 1000 GB
- Tenant service allocation limits: 500 GB, 400 GB, 300 GB

If only a single tenant reaches its allocation limit while the others are well below, the global allocation limit isn't exceeded. Only when several tenants approach their individual allocation limit, the global allocation limit can become a real limit and result in OOMs in all tenants.

Furthermore the following special MDC memory parameters exist:

Parameter	Unit	Default	Validity	Details
global.ini -> [multidb] -> systemdb_reserved_memory	MB	0	>= SPS 12	This parameter allows you to configure a minimal amount of memory (in MB) to be exclusively used by the MDC system database.

29. Which errors indicate memory issues on SAP HANA client side?

Normally memory issues are more likely on SAP HANA server side, but in some scenarios also the SAP HANA client (SAP Note 2393013) can run into a memory bottleneck, e.g.:

Scenario	Client type	SAP Note	Details
ABAP	SQLDBC / ODBC		In SQLDBC / ODBC environments memory issues are reported with errors like: SQL error -9300: no more memory SQL error -10760: Memory allocation failed There can be ABAP short dumps like DBSQL_ALLOCATION_FAILED, DBSQL_DBSL_NO_MEMORY or DBSQL_NO_PERM_MM_MEMORY for similar reasons.
dpagent	JDBC	2400022	The following errors indicate a lack of memory in the data provisioning agent (dpagent): GC overhead limit exceeded (max heap: <heap_mb> MB) Java heap space (failed to allocate <bytes> bytes) (max heap: <heap_mb> MB) The amount of available memory can be adjusted in the dpagent parameter file dpagent.ini via -Xmx switch. See SAP Notes 2399187 and 2737656 for more information.
SAP HANA Studio	JDBC	2073112	The following errors indicate a lack of memory in SAP HANA Studio: GC overhead limit exceeded (max heap: <heap_mb> MB) Insufficient memory for visualization The amount of available memory can be adjusted in the SAP HANA Studio parameter file hdbstudio.ini via -Xmx and -Xms switches. See SAP Note 2159510 for more details.

If you experience these errors, there is usually something wrong with the general memory configuration on client side (operating system or client product like SAP ABAP), e.g. wrong ulimit settings.

30. Can there be fragmentation in the heap memory?

Yes, heap memory can fragment to a certain extent. When an out-of-memory situation happens and the allocated memory is still higher than the used memory, the difference is caused by heap memory fragmentation. You can find related fragmentation information in the out-of-memory dump (SAP Note 1984422), e.g.:

```
Total allocated memory= 760083382272b (707.88gb) Total used memory = 665270861313b (619.58gb)
Heap memory fragmentation: 12
```

Starting with SAP HANA 2.0 SPS04 fragmentation information is also available in column FRAGMENTED_MEMORY_SIZE of monitoring view M_SERVICE_MEMORY and its history HOST_SERVICE_MEMORY. This information is also considered by memory analysis commands available via SAP Note 1969700 (with variant 2.00.040+ or higher).

In general a heap memory fragmentation up to 15 % can be considered as acceptable.

A particularly high, non-reclaimable fragmentation can be a consequence of underlying limitations / configuration issues, e.g. an inadequate setting of /proc/sys/vm/max_map_count. See "Which indications exist that an OOM situation is triggered by the operating system?" for more information.

Be aware that the calculation of the memory fragmentation in trace files can show misleading high values in case of large memory allocation requests, e.g.:

```
Failed to allocate 2565818396904 byte.

...

Heap memory fragmentation: 58% (this value may be high if defragmentation does not help solving
the current memory request)
```

This combination (high 2.4 TB allocation request, high 58 % fragmentation) typically indicates that the high fragmentation value isn't reliable and should be ignored at this point. It is more important to understand and resolve the high memory allocation request.

If you want to check for the current heap memory fragmentation, you can use *SQL: "HANA_Memory_ProcessMemory"* (SAP Note 1969700).

Example:

```
-----------------------------------------------------------------------------------------------
-------
|HOST |PORT |PAL_GB
|ALLOC_GB|HEAP_USED_GB|FREE_GB|FRAG_GB|ALLOC_PCT|HEAP_USED_PCT|FREE_PCT|FRAG_PCT|
-----------------------------------------------------------------------------------------------
-------
|saphana|30003| 176.55| 176.14| 155.34| 0.00| 20.80| 99.77| 87.98| 0.00| 11.78|
-----------------------------------------------------------------------------------------------
-------
```

Effects of different cleanup options on these numbers:

- Internal ad-hoc defragmentation or manual "hdbcons 'mm gc'": Reduction of FRAG_GB, increase of FREE_GB
- Reclaim defragmentation or manual "hdbcons 'mm gc -f'": Minimization of FREE_GB and FRAG_GB
- Reclaim shrink or manual "hdbcons 'resman s'": Reduction of HEAP_USED_GB

Before an OOM is triggered, SAP HANA will always reduce fragmentation as much as possible. It is also possible - but usually not required - to trigger the defragmentation manually as described in "What is memory garbage collection?" above.

31. Which indications exist that an OOM situation is triggered by the operating system?

The following indications exist that an out-of-memory situation is triggered by the operating system and not by SAP HANA:

Symptom	Detail
<service>_<host>.<port>.rtedump.<timestamp>.oom_memory_release.trc dump	This type of SAP HANA dump is only generated in combination with operating system related OOM situations.
[MEMORY_OOM] section in OOM dump: Sum of AB (allocated byte) significantly smaller than GLOBAL_MAX_ALLOCATION_LIMIT "--- precharge ok ---" entries in "Out of memory reasons" overview "Could not return <bytes>b to operating system. This is a configuration problem of your operating system: Please increase	If the sum of allocated memory is smaller than the SAP HANA global allocation limit (and the amount of requested memory is not extraordinary large), the OOM is normally triggered from outside of SAP HANA. In this case you may also see "---

/proc/sys/vm/max_map_count" Other information in OOM dump: Rather small value for /proc/sys/vm/max_map_count (SAP Note 1980196) Value smaller than 100 for SOFTVIRTUALLIMIT in /etc/sysconfig/ulimit in combination with an installed ulimit.rpm package ("rpm -qa | grep ulimit")	precharge ok ---" information in the OOM dump. Reasons can be: Ulimit memory limitation (e.g. due to installed ulimit.rpm package or because of explicit configuration) Inadequate /proc/sys/vm/max_map_count setting (SAP Note 1980196) High soft ulimit setting for stack (SAP Note 2488924) Insufficient physical memory (e.g. due to inadequate SAP HANA memory settings or external software consuming a lot of memory) Address space limit reached (Intel: 128 TB, Power: 16 TB, Power with bigmem: 64 TB); make sure that bigmem flavor is used with Power on SLES 11.x; on SLES >= 12 bigmem is already default Wrong shared memory size information provided by OS for IBM on Power environments (SAP Note 2686011)
/var/log/messages contains messages like: <process> invoked oom-killer Out of memory: Kill process <pid> (hdbindexserver) score <score> or sacrifice child	This OOM killer functionality of Linux is used whenever it runs short on physical memory. In this case processes are terminated in order to reclaim memory.

If you face these symptoms, you can proceed as described in question "Which options exist to reduce the risk of SAP HANA memory issues?" -> "OS configuration" and "Strict NUMA memory binding" above.

32. What is the SAP HANA resource container?

The SAP HANA resource container consists of the row store and heap allocators with information that may be re-used like:

- SAP HANA page cache (Pool/PersistenceManager/PersistentSpace/DefaultLPA/Page)
- Column store tables

It doesn't cover heap areas that can't be re-used - particulary related to SQL statement data processing, e.g.:

- Pool/itab
- Pool/JoinEvaluator/JEAssembleResults
- Pool/parallel/aggregates
- Pool/RowEngine/MonitorView
- Pool/Statistics

- Pool/TableConsistencyCheck

There is no easy approach to identify allocators assigned to the resource container.

You can use *SQL: "HANA_Memory_MemoryObjects"* (SAP Note 1969700) in order to check for the current population of the resource container. The hdbcons command "resman info" (SAP Note 2222218) provides general information related to the current resource container state.

When additional memory is required and not available, SAP HANA can shrink the resource container (e.g. by reduction of certain heap allocators or unloading columns). In this case the database trace (SAP Note 2380176) will contain an entry like the following:

```
Information about shrink at <date> <time> Local: Reason for shrink: Precharge for big block
allocation.
```

The hdbcons command "resman shrink", as e.g. suggested in SAP Note 2301382, only works on the resource container, external allocators can't be shrunk with this command.

Be aware that SUM may perform resource container shrinks during migrations (SAP Note 2685325) that can result in unintended reload activities.

33. How can the types in M_MEMORY_OBJECTS be mapped to allocators?

The object types in monitoring view M_MEMORY_OBJECTS use an individual naming convention. The most important object types can be mapped in the following way:

Type	Allocators / Memory	Details
AttributeEngine/AttributeValueContainerElement	Pool/AttributeEngine* Pool/ColumnStoreTables*	Column store tables
Cache/Hierarchy	Pool/hierarchyBlob	Hierarchy cache
Persistency/Pages/Default	Pool/PersistenceManager/PersistentSpace(0)/DefaultLPA/Page Pool/PersistenceManager/PersistentSpace/DefaultLPA/Page	SAP HANA page cache
Persistency/Pages/RowStore	Shared Memory (allocators Pool/RowStoreTables/* aren't persisted)	Row store

Starting with SAP HANA 2.0 SPS 01 the mapping can be retrieved from monitoring view M_MEMORY_OBJECT_DISPOSITIONS.CATEGORY.

34. In which order are objects unloaded from the resource container?

The unload order of objects in the resource container depends on disposition and unload priority (SAP Note 2127458) settings. A rough mapping is shown in the following table, in general one object type can have portions assigned to different dispositions:

Disposition	Related objects	Parameter	Default
early unload	columns of tables with unload priorities 6 to 9	global.ini -> [memoryobjects] -> disposition_weight_early_unload	100
paged attribute	paged attributes (SAP Note 1871386)	global.ini -> [memoryobjects] -> disposition_paged_attribute	300
(internal) short term	SAP HANA page cache Hierarchy cache	global.ini -> [memoryobjects] -> disposition_weight_short_term	300
lob read lob read small lob write lob write small	disk LOBs	indexserver.ini -> [persistence] -> disposition_lob_read indexserver.ini -> [persistence] -> disposition_lob_read_small indexserver.ini -> [persistence] -> disposition_lob_write indexserver.ini ->	300

		[persistence] -> disposition_lob_write_small	
mid term	temporary nologging retention tables (SAP Note 2800007)	global.ini -> [memoryobjects] -> disposition_weight_mid_term	900
long term	columns of tables with unload priorities 1 to 5 liveCache OMS versions having exceeded min_version_retention_time (SAP Note 2593571)	global.ini -> [memoryobjects] -> disposition_weight_long_term	2700
non swappable	columns of tables with unload priority 0 row store liveCache OMS versions not having reached min_version_retention_time (SAP Note 2593571) main storages in persistent memory (SAP Note 2700084)		0

The disposition weight is divided by the time since the last access of a resource and resources with the smaller resulting values are unloaded first.

Example:

- Column with unload priority 5 last touched 10 hours ago -> disposition result value (based on hours) = 2700 / 10 = 270
- Page in page cache last touched 1 hour ago -> disposition result value (based on hours) = 300 / 1 = 300
- The column has the lower result value (270 vs. 300) and so it is unloaded earlier than the page of the page cache.

In general it is not required to adjust the disposition parameters because the weight factors provide a reasonable overall unload priority, except in a few scenarios:

- In case of critical bugs in the context of unloads it can be useful to increase disposition_weight_long_term and disposition_wait_early_unload (e.g. by factor 10 to 100) in order to make sure that the page cache is unloaded with a higher priority than usual and column unloads are the last resort in case of memory shortage.
- The same applies when you want to minimize column store unloads (e.g. in order to avoid unnecessary reloads or alerts). Be aware that column store unloads can be considered as harmless when only tables with unload priority >= 6 or rarely accessed tables are unloaded. In this case it is neither required nor recommended to adjust the default settings.

Due to a bug with SAP HANA <= 1.00.122.03 it can happen that column unloads happen in an undesired order and critical columns are unloaded earlier than intended (SAP Note 2458491). In this case you can manually unload non-critical columns as a workaround (SAP Note 2127458).

You can use *SQL: "HANA_Memory_Objects_Dispositions"* (SAP Note 1969700) in order to check for current disposition / objects / allocators mappings in a system.

Example:

```
-------------------------------------------------------------------------------------------------
--------------------
|OBJECT_TYPE |DISPOSITION |OBJECT_COUNT|OBJECT_SIZE_GB|SIZE_PER_OBJECT_KB|
-------------------------------------------------------------------------------------------------
```

```
--------------------
|AttributeEngine/AttributeValueContainerElement |LONG_TERM | 1283759| 3442.49| 2811.83|
|Cache/HierarchyCache |SHORT_TERM | 7209| 499.31| 72627.05|
|Persistency/Pages/Default |INTERNAL_SHORT_TERM| 194839| 161.37| 868.47|
|Persistency/Pages/RowStore |NON_SWAPPABLE | 7364608| 116.92| 16.64|
|Persistency/Pages/Default |SHORT_TERM | 1400790| 102.45| 76.69|
|Cache/MdxHierarchyCache |SHORT_TERM | 1225| 40.59| 34744.45|
|AttributeEngine/AttributeValueContainerElement |NON_SWAPPABLE | 1295833| 8.95| 7.24|
|Persistency/Pages/Default |LONG_TERM | 236301| 5.27| 23.42|
|Persistency/Container/VirtualFile |SHORT_TERM | 3505507| 3.13| 0.93|
|Persistency/Pages/Default |TEMPORARY | 3983| 2.65| 699.15|
|Persistency/Pages/Converter/Default |TEMPORARY | 5667| 1.39| 258.69|
-------------------------------------------------------------------------------------------------
--------------------
```

35. Is the SAP HANA memory information always correct?

In general you can rely on the SAP HANA memory information, but the following exceptions exist:

Area	SAP Note	Details
M_CONTEXT_MEMORY		Memory information for granular units like connection and SQL statement are tracked in M_CONTEXT_MEMORY. It can be evaluated via SQL: "HANA_Memory_ContextMemory" (SAP Note 1969700). This information tells you how much statement execution specific memory is currently allocated. This information is usually precise and it is used as basis of memory features like the statement memory limit. The following exceptions exist: SAP Note 2593571 (SAP HANA <= 1.00.122.13, <= 2.00.012.02, <= 2.00.021): Wrong implicit memory booking behavior in context of liveCache procedure calls SAP Note 2603589 (SAP HANA <= 1.00.122.13, <= 2.00.012.02, <= 2.00.022): Allocations in orawstream::reserve are not properly deallocated from context memory. SAP Note 2584388 (SAP HANA <= 1.00.122.14): SQL cache related memory allocations may be accounted for the context memory and so statistics server calls (SAP Note 2147247) can show a high context memory size although at the same time the actual intermediate memory allocation is rather small. SAP Note 2628153 (SAP HANA 1.00.122.16): A wrong memory accounting can result in rising context memory values. SAP Note 2669798 (SAP HANA <= 1.00.122.17): Wrong memory accounting in context of MDS (SAP Note 2670064) SAP HANA <= 2.00.024.02, 2.00.030: Wrong accounting in config::IniParser::parse In the worst case the wrong context memory allocation can result in statement memory limit terminations. As a

		workaround you can set the statement_memory_limit parameter sufficiently high to make sure that it isn't reached by the erroneous context memory value. An implicit memory booking leak can also result in unjustified tracing of database requests as expensive statements if the memory threshold (global.ini -> [expensive_statement] -> threshold_memory) is configured and exceeded by the implicit memory booking (SAP Note 2180165). Existing increased bookings can be cleaned by terminating the related connection (SAP Note 2092196). In case of statistics server sessions a restart is possible based on the description in SAP Note 2584388.
M_EXPENSIVE_STATEMENTS.MEMORY_SIZE	2180165	With SAP HANA <= 1.00.122.13, <= 2.00.012.01, <= 2.00.020 the memory value is incomplete and it is not reliable. With later Revisions it represents the highest memory utilization in an involved service. With SAP HANA <= 1.00.122.21, <= 2.00.024.06 and <= 2.00.034 erroneous -1 values can be reported in some cases (SAP Note 2706472).
M_HOST_RESOURCE_UTILIZATION	2757696	With SAP HANA 2.00.020 - 2.00.024.09 and <= 2.00.037 the column USED_PHYSICAL_MEMORY can contain too large values.
M_SQL_PLAN_CACHE.TOTAL_EXECUTION_MEMORY_SIZE	2124112	With SAP HANA <= 1.00.122.14 the memory consumption of the final close operation is captured, not the peak memory consumption of the actual execution. Starting with SAP HANA 1.00.122.15 the peak memory consumption is properly recorded.

36. How can I get an overview of all recent OOM situations?

Trace files may not cover all OOM situations because a trace is only written after the configured oom_dump_time_delta (default: 1 day) is exceeded. Instead you can find an overview of OOM situations in monitoring view M_OUT_OF_MEMORY_EVENTS (SAP HANA 1.0 >= SPS 12), the related statistics server history GLOBAL_OUT_OF_MEMORY_EVENTS or alternatively via *SQL: "HANA_Memory_OutOfMemoryEvents"* (SAP Note 1969700).

Example:

```
------------------------------------------------------------------------------------------------
------------------------------------------------------------- |OOM_TIME |HOST |PORT |CONN_ID
|STATEMENT_HASH |MEM_REQ_GB|MEM_USED_GB|MEM_LIMIT_GB|EVENT_REASON |TRACEFILE_NAME| -------------
------------------------------------------------------------------------------------------------
------------------------------------------------ |2018/01/21 14:56:37|saphana6|30003|
509002|8c9a904596ef7297c18047ae899593d4| 7.28| 199.35| 200.00|GENERIC_COMPOSITE_LIMIT| |
|2018/01/21 14:56:38|saphana6|30003| 509002|8c9a904596ef7297c18047ae899593d4| 7.27| 199.38|
200.00|GENERIC_COMPOSITE_LIMIT| | |2018/01/21 14:56:40|saphana6|30003|
509002|8c9a904596ef7297c18047ae899593d4| 7.26| 199.45| 200.00|GENERIC_COMPOSITE_LIMIT| |
|2018/01/21 14:56:43|saphana6|30003| 509002|8c9a904596ef7297c18047ae899593d4| 7.27| 199.57|
200.00|GENERIC_COMPOSITE_LIMIT| | |2018/01/21 14:56:44|saphana6|30003|
509002|8c9a904596ef7297c18047ae899593d4| 0.50| 199.96| 200.00|GENERIC_COMPOSITE_LIMIT| |
|2018/01/22 14:14:19|saphana5|30003| 408089|ea8afb5aed39f133e5f593dfaed1828b| 0.00| 200.00|
200.00|GENERIC_COMPOSITE_LIMIT| | |2018/01/23 17:28:35|saphana6|30003|
508413|0450975123f2a81eb26a1ebc06f819cf| 3.21| 197.64| 200.00|GENERIC_COMPOSITE_LIMIT| |
|2018/01/24 11:37:24|saphana6|30003| 416809|d589b47003b8db3caf9425ebfaf5b72e| 11.06| 189.43|
200.00|GENERIC_COMPOSITE_LIMIT| | ----------------------------------------------------------------
-----------------------------------------------------------------------------------------------
```

See SAP Note 2088971 for possibilities to control the numer of records in M_OUT_OF_MEMORY_EVENTS.

37. Is SAP HANA aware about dynamic memory changes?

If you adjust the amount of physical memory while SAP HANA is up and running, SAP HANA won't automatically consider the new size. To avoid issues you can manually adjust memory related parameters like global_allocation_limit and allocationlimit or synchronize memory adjustments with times of SAP HANA restarts.

38. Are all SAP HANA services part of the memory management?

No, not all SAP HANA services (SAP Note 2477204) are covered by the memory management. Exceptions are:

- daemon
- esserver
- etsserver
- rdsyncserver
- streamingserver
- xscontroller
- xsexecagent
- xsuaaserver

As a consequence values in memory related monitoring views may be missing or having unexpected values (e.g. -1 for the process allocation limit).

39. Is there a specific shared memory configuration required?

It is important that the utilization of shared memory isn't limited on operating system side, otherwise you may face various trouble in context of row store load or growth, e.g.:

- Terminations with error "132: transaction rolled back due to unavailable resource" (SAP Note 2399990)
- Trace file entries with "ShmSystem::create - No space left on device" (SAP Note 2380176)
- Indexserver crash during startup with "shared memory allocation failed" (SAP Note 2534844)
- Indexserver emergency dump during startup in ptime::PTimeFactory::startMaster
- *SQL: "HANA_TraceFiles_MiniChecks"* (SAP Note 1969700) reports check ID T0319 ("Shared memory: No space left on device")

To avoid trouble you should make sure that the shared memory settings on OS level are set to sufficiently large values (SAP Notes 941735, 2534844):

- kernel.shmmni = 32768 (segments)
- kernel.shmmax >= 1000000000000000 (byte)
- kernel.shmall >= 1000000000000000 (byte)

By default, the Linux distributions already set extremely large values for kernel.shmmax and kernel.shmall. We recommend that you keep these values unchanged.

Changes to these settings are immediately taken into account, no restart / reboot is required.

You can check the current values with the following Linux command (with <parameter> = kernel.shmmni, kernel.shmmax or kernel.shmall):

```
sysctl <parameter>
```

40. How can memory activities be traced?

A database trace for memory operations can be configured with the following parameter (SAP Note 2380176):

```
global.ini -> [trace] -> memory
```

Per default it is already activated on "info" level, so only "debug" may provide more information. Vice versa it can be useful in some scenarios to reduce the tracing, e.g. to level "error" as suggested in SAP Note 2694985.

41. Where do I find information about persistent memory and the fast restart option?

Persistent memory (SAP HANA >= 2.00.035) and the fast restart option (SAP HANA >= 2.00.040) provide the possibility to retain column store main storages in memory across SAP HANA and / or server restarts. See SAP Note 2700084 for more details.

42. What is the SAP HANA free memory cache?

Starting with Release 2.0 SPS 03 SAP HANA administers an own free memory cache in order to reduce page faults on operating system side. Starting with SAP HANA 2.0 SPS 04 you can determine its size via monitoring view M_HEAP_MEMORY_AREAS (AREA = 'FreeMemoryCache') or *SQL: "HANA_MemoryOverview"* ("HANA free memory cache") available via SAP Note 1969700.

Usually it works fine and there is no need for any configuration. In some cases the cache can have an adverse performance effect:

- SAP HANA 2.0 SPS 03 - SPS 04: Degrading system performance after SAP HANA or OS kernel upgrade (SAP Note 2857553)
- Many threads in call stacks MemoryManager::PrechargeBase::remapBlocksFromLocalFreeListWithoutUpdateValues and / or MemoryManager::PrechargeBase::remapMemory
- Many threads in call stacks indicating memory intensive operations like IndexVectorAligned::get,introsortLoopAux, partitionFind or ValueArray::init

These scenarios can always be indications for other issues, e.g. inadequate NUMA configuration (SAP Note 2470289) or a general issue in the memory area. In some situations it can be useful to disable the SAP HANA memory cache. This can be achieved by disabling the memory cache with an environment parameter:

```
daemon.ini -> [daemon] -> environment = HDB_MEMORY_CACHE=d
```

This parameter has to be set on SAP HANA node level, e.g.:

```
ALTER SYSTEM ALTER CONFIGURATION ('daemon.ini', 'HOST', '<hana_node>') SET ('daemon',
'environment') = 'HDB_MEMORY_CACHE=d' WITH RECONFIGURE
```

SAP HANA needs to be restarted for this change to take effect.

Attention: Be aware that disabling the SAP HANA memory cache is normally neither necessary nor recommended, so you should only do it in special scenarios where other optimizations aren't possible and you see a clear benefit from the disabled cache.

43. What is the SAP HANA memory profiler?

the SAP HANA memory profiler is available with SAP HANA >= 2.0 SPS 04 and it allows activating detailed tracing of memory allocations. It can either be controlled via SAP HANA Cockpit (SAP Note 2800006) as described in Analyzing Memory with the Memory Profiler or via the following explicit SQL commands:

Command	Details
ALTER SYSTEM CLEAR MEMORY PROFILER	Clear previous memory profiler results
ALTER SYSTEM LOAD MEMORY PROFILER [FROM '<profile>'] INTO TABLES <prefix> [WITH REPLACE]	Load memory profiler results into tables with names starting with "<prefix>_" FROM '<profile>': Use specific profile name <profile> WITH REPLACE: Overwrite existing target tables, otherwise it will fail with "profile already loaded" in case the target tables already exist
ALTER SYSTEM START MEMORY PROFILER [TO '<profile>'] [WITH REPLACE] [SAMPLING INTERVAL <interval_ms>] [DURATION <duration_s>] FOR ALL ALLOCATORS [EXCEPT FOR ALLOCATOR '<allocator1>', ..., '<allocatorN>'] [FOR ALLOCATOR WITH CALLSTACK '<allocator1>', ..., '<allocatorM>'] | FOR ALLOCATOR '<allocator1>', ..., '<allocatorN>' [FOR ALLOCATOR WITH CALLSTACK '<allocator1>', ..., '<allocatorM>'] | FOR ALLOCATOR WITH CALLSTACK '<allocator1>', ..., '<allocatorM>'	Start of memory profiler TO '<profile>': Use specific profile name <profile> WITH REPLACE: Overwrite existing profiler data SAMPLING INTERVAL: Sampling interval (ms), default: 10 ms, in context of call stack generation it is recommended to increase it to >= 100 ms DURATION: Profiling duration (s) FOR ALL ALLOCATORS: Activate profiling for all allocators EXCEPT FOR ALLOCATOR: Exclude dedicated allocators from profiling FOR ALLOCATOR: Activate profiling for specific allocators FOR ALLOCATOR WITH CALLSTACK: Activate callstack profiling for specific allocators
ALTER SYSTEM STOP MEMORY PROFILER	Stop of memory profiler

You can add "AT '<host>:<port>'" to all commands to restrict the execution to a specific host and port.

The system privilege MEMORY PROFILER ADMIN is required for these operations.

The raw trace details are written to a trace file with naming convention <service>_<host>.<port>.**memory**.trc or (in case TO '<profile>' is specified) <service>_<host>.<port>.<profile>.**memory**.trc.

The following errors can happen in context of memory profiling:

Error	Details
2: general error: memory profile not found	You try to process a memory profile that doesn't exist, yet. For example, you try to execute LOAD before you have executed STOP for the current memory profiler run.
2: general error: profile already loaded	This termination happens when you try to load the memory profile data into a set of tables with a prefix that already exists. Either drop the previous tables or use the WITH REPLACE option.
129: transaction rolled back by an internal	You try to start the memory profiler although it is already running. It needs to be stopped before it can be restarted.

error: Memory Profiler already started.	
129: transaction rolled back by an internal error: profile already exists	You try to generate a memory profile for which the memory trace file aready exists. You can overwrite the existing one by choosing the WITH REPLACE option. Alternatively you can delete the previous memory trace file, e.g. via: ALTER SYSTEM CLEAR TRACES ('<host>', '<memory_profiler_trace_file>')

You can use *SQL: "HANA_Memory_MemoryProfiler"* (SAP Note 1969700) in order to evaluate loaded memory profiler data.

44. How can allocations in Pool/Statistics be analyzed and optimized?

The heap allocator Pool/Statistics contains statistical information for different contexts. In order to understand the main space contributors and the steps forward it is useful to display the top contributors in the first place using the blocklist option of hdbcons (SAP Note 2222218):

```
hdbcons 'mm bl -t Pool/Statistics'
```

Be aware that you have to execute it on the host and for the service that suffers from the high Pool/Statistics size.

The output provides the main contributors in terms of memory consumption sorted top-down. Always check the top-most entries first. The following known scenarios exist:

Scenario	Top blocklist contributors	Details
Many connections / cached statements	Execution::ContextAllocator::init ImplicitStatementMemoryBooking ltt::allocator_statistics::setCompositeLimit MemoryManager::LimitInfo::LimitInfo MemoryManager::MemoryCounter::MemoryCounter MemoryManager::PoolAllocator::PoolAllocator MemoryManager::StripedAllocator::allocateStriped	These allocators are linked to context memory details. The size mainly depends on the following factors: Number of records in M_CONTEXT_MEMORY (which is closely linked to the number of SQL connections and cached statements) Number of (logical) CPUs Activation of special features like memory tracking or statement memory limit For example, 578 CPU threads, 4.5 million entries in M_CONTEXT_MEMORY and activated memory tracking and statement memory limit can result in an allocator size of 300 GB. The amount of entries in M_CONTEXT_MEMORY among others depends on the amount of database requests cached on client side. In ABAP environments this is controlled by parameter dbs/hdb/stmt_cache_size (default: 1000 statements per connection). You can reduce it in order to minimize the M_CONTEXT_MEMORY entries and the Pool/Statistics size (SAP Note 2532199). In a real-life scenario the size of Pool/Statistics reduced by factor 4 after having reduced the value from 1000 to 100. Be aware that a reduction of this setting can increase the parsing activities, so you should monitor the performance effects. See SAP Note 2124112 ("Can there be also statement caches on client side?") for more information related to the ABAP client statement cache. See SAP Note 2600030 for best practice

		recommendations for parameter dbs/hdb/stmt_cache_size. With SAP HANA >= 2.00.040 it is possible to reduce the size of Pool/Statistics required for context memory information by storing information per NUMA node rather than CPU core. This can be controlled by setting the following SAP HANA parameter to 'numa': global.ini -> [memorymanager] -> composite_statistics_striping The following values are possible: auto: internal decision, currently identical to 'core' core: core based statistics as before, i.e. higher memory requirements with optimal performance numa: NUMA node based statistics, i.e. reduced memory requirements with the risk of performance reductions When setting this parameter to numa it can happen that performance and CPU overhead is visible in context of module MemoryManager::StripedAllocator::allocateStriped. In this case you can disable internal striping with the following parameter (SAP Note 2100040): global.ini -> [memorymanager] -> statistics_type = 7
Transactional LOBs	Execution::ContextAllocator::initImplicitStatementMemoryBooking ltt::allocator_statistics::setCompositeLimit MemoryManager::LimitInfo::LimitInfo MemoryManager::MemoryCounter::MemoryCounter MemoryManager::PoolAllocator::PoolAllocator MemoryManager::StripedAllocator::allocateStriped	Pool/Statistics can also grow in context of many prepared statements not being closed in context of transactional LOBs and JDBC (SAP Notes 2220627, 2711824). Consider to deactivate transactional LOBs.
Problem with allocator sharing	MemoryManager::StripedAllocator::allocateStriped	When SQL: "HANA_Memory_ContextMemory" (SAP Note 1969700) reports a high amount of entries for Connection/.../Pool/RowEngine/Session with SAP HANA <= 1.00.122.30, <= 2.00.037.05 and <= 2.00.046, it can be caused by the SAP HANA bug described in SAP Note 2910730. As a temporary workaround you can disable allocator sharing by setting the following SAP HANA parameter: global.ini -> [memorymanager] -> enable_sharing_allocator_for_implicit = false
Many conditional variables	Synchronization::CondVariable::CondVariable	This module indicates that space is required for conditional variable locks. In this case you can use the following hdbcons command (SAP Note 2222218) to identify existing conditional variables (attention: output can be large in case of many

		locks): hdbcons 'statreg print --all -h -n M_CONDITIONAL_VARIABLES' In case the lock ptime::MixedJoinInfo::sync is dominant, you can implement the following SAP HANA parameter as a workaround: indexserver.ini -> [sql] -> native_mixed_join_enabled = false SAP will also provide a fix for this issue with a future SAP HANA Revision level.
Many read write locks	Synchronization::ReadWriteLock::ReadWriteLock Synchronization::FastReadSlowWriteLock::allocateReaderItems	With SAP HANA <= 1.00.122.12 and 2.0 SPS 00 read write locks were stored in Pool/Statistics. With newer SAP HANA Revisions the dedicated heap allocator Pool/FRSWLockAllocator is used for that purpose. See Pool/FRSWLockAllocator for troubleshooting details.
Many heap allocators		If there are many records in M_HEAP_MEMORY (> 100000), you can check for the most frequent heap allocators using SQL: "HANA_Memory_TopConsumers" (DATA_SOURCE = 'CURRENT', AREA = 'HEAP', ORDER_BY = 'COUNT') of SAP Note 1969700.

45. What is a good table memory share?

It is recommended that tables (and related objects like indexes) don't use more than 50 % of the available memory so that sufficient memory is available for other important areas like intermediate result sets, caches and statistics. The more the table footprint exceeds the 50 % limit the higher is the probability of out-of-memory events or hiccups due to memory reclaims and resource container shrinks.

Keywords

SAP HANA memory heap allocator table row column store oom out of memory

Products

SAP HANA, platform edition all versions

This document refers to

SAP Note/KBA	Title
2800008	FAQ: SAP HANA Fulltext Indexes
2800007	FAQ: SAP HANA Temporary Tables
2800006	FAQ: SAP HANA Cockpit

2799997	FAQ: SAP HANA Native Storage Extension (NSE)
2737656	How to increase DP Agent memory
2700084	FAQ: SAP HANA Persistent Memory
2685325	SUM Executes HDB_SHRINK during downtime phases
2670064	FAQ: SAP HANA Multi-Dimensional Services (MDS)
2667371	HANA throws out of memory dump while accessing backup catalog
2600076	FAQ: SAP HANA Inverted Individual Indexes
2600030	Parameter Recommendations in SAP HANA Environments
2599949	FAQ: SAP HANA Extended SQL Executor (ESX)
2593571	FAQ: SAP HANA Integrated liveCache
2573880	FAQ: SAP HANA Full System Info Dump
2570371	FAQ: SAP HANA Execution Engine (HEX)
2520774	FAQ: SAP HANA Performance Trace
2506811	FAQ: SAP HANA Dynamic Result Cache
2502256	FAQ: SAP HANA Caches
2477204	FAQ: SAP HANA Services and Ports
2470289	FAQ: SAP HANA Non-Uniform Memory Access (NUMA)
2467292	memAllocSystemPages failed with rc=12 - Cannot allocate memory
2453348	Out of Memory Occured with Large Pool/planviz/ and Pool/RowEngine/QueryCompilation
2416490	FAQ: SAP HANA Data Aging in SAP S/4HANA
2400022	FAQ: SAP HANA Smart Data Integration (SDI)
2400005	FAQ: SAP HANA Persistence
2399993	FAQ: SAP HANA Fast Data Access (FDA)
2399990	How-To: Analyzing ABAP Short Dumps in SAP HANA Environments
2393013	FAQ: SAP HANA Clients
2388483	How-To: Data Management for Technical Tables
2380176	FAQ: SAP HANA Database Trace
2375917	How-To: Converting SAP HANA VARBINARY columns to LOB
2370588	S/4 migration job causes an Out Of Memory during the MUJ step on a HANA System

2349081	Datavolume increase following an upgrade to SPS09 or higher
2340450	FAQ: SAP HANA Table Replication
2336344	FAQ: SAP HANA Static Result Cache
2302903	HANA PlanViz "Memory Allocated" figure is higher than the statement memory limit
2242507	HANA out of memory problem while using Smart Data Access
2222718	Troubleshooting HANA High Memory Consumption - Guided Answers
2222277	FAQ: SAP HANA Column Store and Row Store
2222250	FAQ: SAP HANA Workload Management
2222218	FAQ: SAP HANA Database Server Management Console (hdbcons)
2222200	FAQ: SAP HANA Network
2220627	FAQ: SAP HANA LOBs
2180165	FAQ: SAP HANA Expensive Statements Trace
2180119	FAQ: SAP HANA Smart Data Access
2177064	FAQ: SAP HANA Service Restarts and Crashes
2175606	HANA: How to set allocation limit for tenant databases
2169283	FAQ: SAP HANA Garbage Collection
2160391	FAQ: SAP HANA Indexes
2159510	HANA Studio - The Load graph or PlanViz do not display
2159014	FAQ: SAP HANA Security
2154870	How-To: Understanding and defining SAP HANA Limitations
2147247	FAQ: SAP HANA Statistics Server
2143679	How-To: Removing Primary Keys of SAP HANA Statistics Server Histories
2142945	FAQ: SAP HANA Hints
2127458	FAQ: SAP HANA Loads and Unloads
2124112	FAQ: SAP HANA Parsing
2122650	HANA oom trace file dumps with 'Composite limit violation (OUT OF MEMORY) occurred' in HANA SPS 08 and higher
2119087	How-To: Configuring SAP HANA Traces
2116157	FAQ: SAP HANA Consistency Checks and Corruptions
2112604	FAQ: SAP HANA Compression

2109355	How-To: Configuring SAP HANA Inverted Hash Indexes
2101244	FAQ: SAP HANA Multitenant Database Containers (MDC)
2100040	FAQ: SAP HANA CPU
2100009	FAQ: SAP HANA Savepoints
2092196	How-To: Terminating Sessions in SAP HANA
2088971	How-To: Controlling the Amount of Records in SAP HANA Monitoring Views
2081869	How to handle HANA Alert 64: 'Total memory usage of table-based audit log'
2081591	FAQ: SAP HANA Table Distribution
2081473	HANA Resident Memory : High Memory Usage
2073964	Create & Export PlanViz in HANA Studio
2073112	FAQ: SAP HANA Studio
2057046	FAQ: SAP HANA Delta Merges
2050579	How to handle HANA Alert 68: 'total memory usage of row store'
2044468	FAQ: SAP HANA Partitioning
2000003	FAQ: SAP HANA
2000002	FAQ: SAP HANA SQL Optimization
2000000	FAQ: SAP HANA Performance Optimization
1999998	FAQ: SAP HANA Lock Analysis
1999993	How-To: Interpreting SAP HANA Mini Check Results
1999930	FAQ: SAP HANA I/O Analysis
1999880	FAQ: SAP HANA System Replication
1998599	How-To: Analyzing high SAP HANA Memory Consumption due to Translation Tables
1984422	How-To: Analyzing SAP HANA Out-of-memory (OOM) Dumps
1977269	How to handle HANA Alert 45: 'Check memory usage of main storage of column-store tables'
1977268	How to handle HANA Alert 40: 'Total memory usage of column-store tables'
1977207	How to handle HANA Alert 55: Columnstore unloads
1977101	How to handle HANA Alert 12: 'Memory usage of name server'
1900257	How to handle HANA Alert 43: 'Memory Usage of Services'
1899511	How to handle HANA Alert 44 'Licensed Memory Usage'

1898317	How to handle HANA Alert 1: ‘Host physical memory usage’
1862506	HANA: Statisticsserver runs out of memory (OOM) as of SPS05
1847202	Error "400 Bad Request" when executing EPM Add-in report with a big amount of dimension members to be retrieved - BPC NW
941735	SAP memory management system for 64-bit Linux systems
2910004	Increase Disk Usage After Performing DDL Operation on Table With Packed LOB Column
2866563	Memory Leak Caused by Mishandling of Temporary Index Used by Calculation Operation After Query Cancellation
2857553	Overall System Performance Degrading After HANA or OS Kernel Upgrade
2843100	Memory Leak in Pool/PersistenceLayer
2839027	Memory Usage in Memory Allocator Pool/RowEngine/SQLPlan Much Higher Than Configured in [sql] plan_cache_size
2818480	Temporarily High Memory Usage in Pool/PersistenceManager/Backup/Superblock During Resumed Data Shipping
2815963	Temporarily High Memory Consumption in Pool/RowEngine/GlobalHeap and Pool/KernelSentinel
2808956	Increased Used Memory Size due to Pool/L/llang/Debuggee
2789785	Memory leak while calling virtual procedure.
2789255	Automatic Online Row Store Reorganization
2785533	Using SQL Commands to get Recommendations for the SAP HANA Native Storage Extension (NSE) Advisor
2780510	SAP HANA 2.0 SPS 03 Database Maintenance Revision 037.01
2757696	HANA Alert 1: 'Host physical memory usage' shows wrong information
2749005	Continuously increasing memory usage seen in dpserver while doing replication.
2746957	Out of Memory at ESX::UnionAll::open
2746759	SAP HANA 2.0 SPS 03 Database Revision 037
2740826	Indexserver Crash or OOM at AabapSysTimezone
2731521	Index Server Crash at JoinEvaluator::accessHashMapRows() due to Customized min_segment_size Parameter
2711824	High Number of Prepared Statements Causing High Usage of Memory Allocator Pool/Statistics
2707020	Disk Size of Columnstore Table Containing Packed LOBs (also called MidSize LOBs) Does not Decrease After Truncate
2706472	Sporadic -1 Value in MEMORY_SIZE Column of M_EXPENSIVE_STATEMENTS Monitoring View When Memory Resource Tracking And Expensive Statement Tracing Are Enabled

2694985	Many Warning Messages "Could not get some memory usage statistics from Cgroup, fetching from /proc/meminfo file" in HANA Traces
2686011	Accounting for Shared Memory Size is Wrong on IBM Power
2678164	Default Configuration of Parameter [system_replication] logshipping_async_buffer_size Increased for Indexserver
2669798	Query Execution Leads to an Out of Memory Situation Though Statement Memory Limit is Set
2669159	Compilation of Query With Several Long IN Lists Fails With High Memory Usage in Pool/RowEngine/QueryCompilation
2643641	DPServer Memory Utilization
2637828	Memory Leak on Pool/malloc/libhdbbasement.so When Collecting Performance Trace/Planviz/Plan Trace with Function Profiler
2629536	Unexpected Composite OOM Errors Caused by Setting Total Statement Memory Limit
2628153	Unexpected Composite Out of Memory Event Occurs Frequently
2624305	Potential Memory Leakage on Pool/malloc/libhdbcswrapper.so
2612205	HANA Indexserver Cannot Load Row Store Tables Because of OOM
2612022	Increased Memory Allocator Size After Distributed Query Execution Failed due to OOM
2603589	Composite OOM in orawstream::reserve
2601475	Memory Leak in Pool/malloc/libhdbcsapi.so When Running Enterprise Search Queries
2599658	Increased Version Count or Data Volume Size and Memory Consumption Increase due to Dangling Transtoken
2597818	Memory Leak in Pool/ESX When Using PlanViz Execution
2588395	Erroneous Accounting of Shared Memory in Multitenant Database Container Systems Running in High Isolation Level on Linux
2584388	High Memory Usage in Allocator Connection/XXXXXX/Statement/YYYYYYYY/IMPLICIT by User _SYS_STATISTICS
2583148	Higher garbage memory build up in SAP HANA due to TMA application
2573738	Rowstore Versions are not Collected on System Replication Target Site When Using Operation Mode Logreplay
2547516	Consistency Check Execution Causes Growth of Pool/malloc/libhdbbasement.so
2542700	DPServer memory utilization continues to climb when processing cluster tables
2535110	Memory Leak on Pool/parallel/compactcol and Pool/parallel/aggregates or Pool/itab
2534844	Indexserver Crash During Startup due to Insufficient Shared Memory Segment
2533352	Memory Leak on "Pool/JoinEvaluator/JERequestedAttributes/Results"

2532199	Optimization of the HANA Memory Allocator Pool/Statistics Usage
2527251	Memory Leak in Pool/RowEngine/QueryCompilation
2517443	Filter push down missing for TREXviaDBSL calls on Hana native calculation view when FEMS are used
2497016	Pages Belonging to Cold Partitions Created With Paged Attribute Are Not Unloaded by The Resource Manager if They Are Pinned by an Inverted Index
2488924	Linux: Recommended values for maximum stack size of processes
2458491	Unloads of Recently Columns Despite Older Columns Could be Evicted
2415279	How-To: Configuring SAP HANA for the SAP HANA Extension Node
2405763	SAP HANA DB: Log Replay on HSR Secondary Site Hangs
2403124	Optimization of HANA Page Cache Usage
2399187	Out of Memory conditions in DP Agent (Java heap space error message, GC overhead limit exceeded error message)
2398507	Memory in Pool/ICT is Constantly Increasing and Does not get Released
2376550	HANA CONCAT attributes for BW Column Views: Correction for SID-Tables
2371445	SAP HANA SPS 12 Database Maintenance Revision 122.03
2312983	Memory leak in Pool/parallel/aggregates when querying on distributed environment
2146989	SAP HANA: High Number of Persistent Pages of Type UnifiedTableMVCC
2014148	Guidelines for Using the Query Result Cache
1993128	SAP HANA: column store table unloads and unloading behavior of Memory Objects Container
1980765	Operations with columns containing only one value may lead to wrong data
1969700	SQL Statement Collection for SAP HANA
1903576	SAP HANA DB: Additional Main Memory in Exceptional Cases
1900823	SAP HANA Storage Connector API
1871386	SAP HANA: Paged Attributes
1865554	MDX: Access type F4 help / improved error update
1813245	SAP HANA DB: Row store reorganization
	Analyzing Memory with the Memory Profiler
	SAP HANA Troubleshooting and Performance Analysis Guide
	SAP HANA Administration Guide
	ABAP Sourcecode Search

	Simplification List for SAP S/4HANA
	Information Lifecycle Management