

SAP How-to Guide

Business Analytics

SAP HANA™ Appliance

How To Delta Merge for SAP HANA and SAP BW powered by SAP HANA

Applicable Releases:

SAP HANA 1.0 SPS 05 and SAP BW powered by HANA 7.3 SP6 or higher

Version 2.2

January 2015



The Best-Run Businesses Run SAP™

© Copyright 2015 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase, Inc. Sybase is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.

This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.

SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.

SAP "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

Disclaimer

Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressly prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP's Support Services and may not be modified or altered in any way.

Document History

Document Version	Description
2.20	Updates for BW 7.40 functionality and new cost functions
2.10	Updated the cost functions and the process chain type for delta merge
2.00	Added details on delta merge and merge types
1.00	First official release of this guide



Typographic Conventions

Type Style	Description
<i>Example Text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation
Example text	Emphasized words or phrases in body text, graphic titles, and table titles
Example text	File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
Example text	User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

Icons





Icon	Description
	Caution
	Note or Important
	Example
	Recommendation or Tip

Table of Contents

1.	Business Scenario	1
2.	Background Information	1
3.	Prerequisites	1
4.	Column Store	2
5.	Delta Merge	3
5.1	Merge Process	3
5.2	Merge Types	4
5.2.2	Smart Merge	5
5.2.3	Other	5
5.3	Merge Monitor	7
5.4	Token Concept	7
5.5	Configuration	7
5.5.1	Cost Function	8
5.5.2	Merge Token Configuration	10
5.6	SQL Syntax	11
6.	Monitoring	12
6.1	Merge History	12
6.2	Alerts	14
7.	SAP BW Specifics	17
7.1	Differences by Object Type	17
7.1.1	DTP Trigger Setting for Invoking “Smartmerge” Hint	18
7.1.2	Process Type to Trigger “Smartmerge” Hint	18
7.1.3	BW Application Log entries for Delta Merge Trigger	19
7.2	Review Automerge Setting by Table	19
8.	Appendix	21
8.1	Appendix A - Temporal Tables for Data Store Object Change Log	21
8.2	Appendix B - Cost Function Examples	21

1. Business Scenario

The implementation and operation of SAP HANA requires a solid understanding of the delta merge process in the database. Understanding this will allow administrators to optimize data loading and can in some cases help improving the execution performance of query processing in SAP HANA or any application (e.g. SAP BW). Delta merge processing can consume a non-trivial amount of system resources) and therefore needs to be understood in order to manage any system powered by SAP HANA.

2. Background Information

This document provides an overview of the SAP HANA delta merge process and the specific operations that affect applications such as SAP BW powered by SAP HANA. In order to provide insight on this topic, the document will cover the following:

- Updating the column store
- Delta merge concept
- Delta merge process
- Monitoring the merge process
- Configuring the merge process
- Implications for applications like SAP BW powered by SAP HANA



Caution

- Within this document references are made to configuration. There is no recommendation in this document to change any of the default settings!

3. Prerequisites

Readers who will benefit most from this document are those who have had HANA technical training and/or reviewed RKT material

- SAP HANA 1.0 SPS5, SAP BW powered by HANA 7.3 SP6 or higher

Provide information about:

- Relevant SAP Notes
 - SAP Note 1663501 - <https://service.sap.com/sap/support/notes/1663501>
- Additional background/starting documentation (also provide a link)
 - http://help.sap.com/content/hana/overview/hana_overview.htm
- Required/recommended expertise or prior knowledge
 - SAP HANA technical training
 - SAP HANA RKT documentation reviewed

4. Column Store

A column store table is comprised of two index types, for each column a main index and a delta index. The delta storage is optimized for write operations and the main storage is optimized in terms of read performance and memory consumption. The use of the delta tables addresses the performance issues of loading directly to compressed columns.

As described in the Figure “Read and Write Operations on a Column Store Table” below, read operations are performed on both parts whereas write operations only affect the delta part. In order to optimize query execution performance of the system and to ensure optimum compression, the system needs to transfer the data from the delta part into the main part. This process is called delta merge.

This document is intended to give more insight on the merge process and the impact on SAP BW powered by SAP HANA administration tasks.

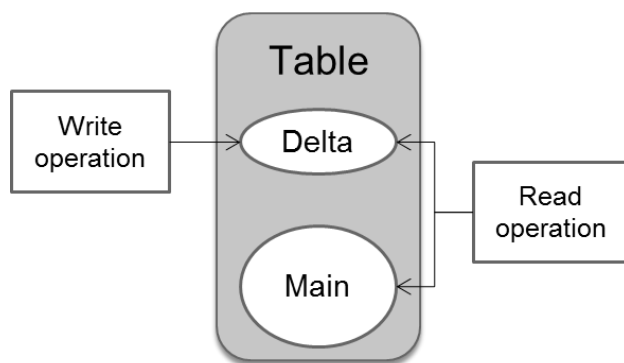


Figure: Read and Write Operations on a Column Store Table



Note

- There are different states a column store table can be in
 - Unloaded - none of the column store data is loaded to main memory.
 - Partly loaded - parts of the column store data are loaded to main memory e.g.: only a few columns recently used in a query.
 - Fully loaded - all data of the column store is loaded into main memory



Note

- For this document, it is sufficient to handle the delta part as one object and the main part as one object. In the subsequent sections, this document will refer to them as delta storage and main storage.

5. Delta Merge

Write operations are only performed on the delta storage. In order to transform the data into a format that is optimized in terms of memory consumption and read performance, it has to be transferred to the main storage. This is accomplished by a process called delta merge. This section is intended to give a better understanding of how this happens and when.

5.1 Merge Process

1. The Figure “Delta Merge Process” below describes the different states of a merge process, which objects are involved and how they are accessed.

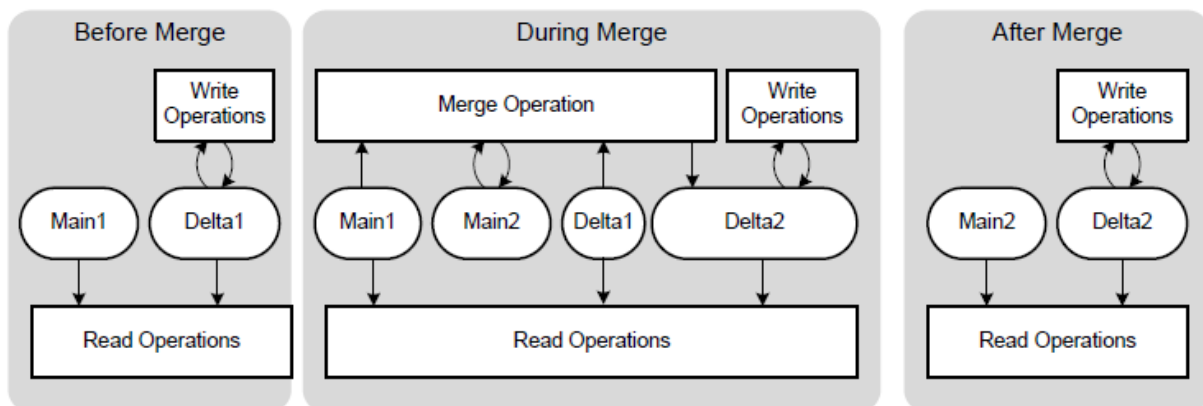


Figure: Delta Merge Process

Before the merge operation:

All write operations go to storage Delta1 and read operations read from storages Main1 and Delta1.

During the merge operation:

While the merge operation is running, all changes go into the second delta storage, Delta2. Read operations read from the original main storage, Main1, and from both delta storages, Delta1 and Delta2. Uncommitted changes in Delta1 are copied to Delta2. The content of Main1 and the committed entries in Delta1 are merged into the new main storage, Main2.

After the merge operation:

After the merge, Main1 and Delta1 storages are deleted.

2. The optimization of the compression algorithm is the second step of the delta merge process. After successful completion of the merge, this step calculates the optimized compression parameters. The system immediately reloads any column into memory if compression has changed.

Note

- With this double buffer concept, the table only needs to be locked for a short duration starting just before open transactions are moved to delta2 and ending just after the storages are “switched”.

Caution

- As we can see in Figure “Delta Merge Process” above, the minimum memory requirement for delta merge includes the current main size + future main size + current delta + some additional memory.

It is important to understand, even if a column store table is unloaded or partly loaded, the whole table will be loaded into memory in order to perform the delta merge.

5.2 Merge on Partitioned Tables

Executing a delta merge on partitioned tables can have a positive impact on the memory consumption during merge operation and the performance. Every partition of a table is treated internally as a standalone table with its own data and delta store.

When it comes to executing the delta merge, only the affected partitions of a table will be treated by the merge operation. As described before, the whole table has to be duplicated during the merge. For partitioned tables therefore, the amount of needed main memory during the merge is reduced, depending on the size of the partition.

As the merge duration mainly depends on the size of the merged table in main memory, merging a partition only also affects the merge runtime positively.

Caution

Before a table is (re-)partitioned, a delta merge is executed. Therefore you have to partition tables in time to not run out of memory during a merge for huge tables.

5.3 Merge Types

One can put the different merge types in two categories for instance, the ones that are automatically executed by the system via a process called mergedog and the ones that can be triggered manually, either through an application (e.g. BW) or by a user sending an SQL statement to HANA.

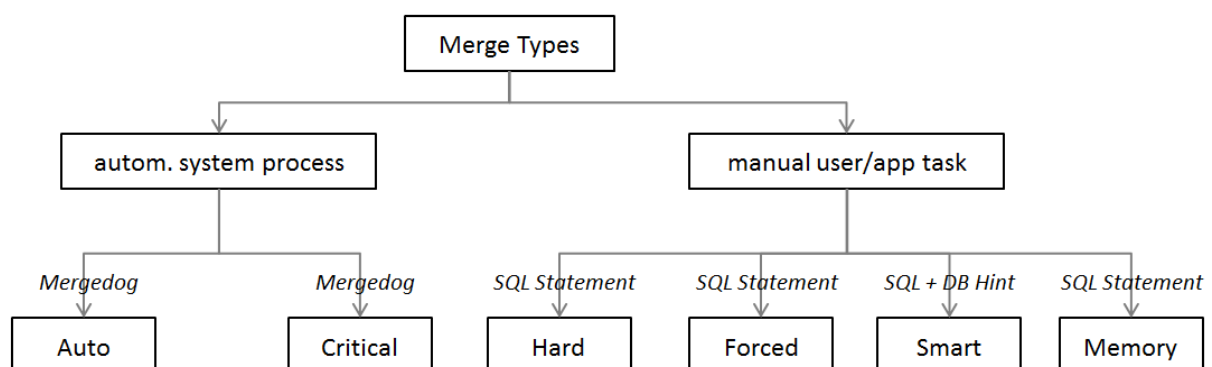


Figure 1: The different merge types in HANA

5.3.1 Auto Merge (default setting)

There are different ways to initiate a merge process. The standard method for initiating a merge in SAP HANA is via auto merge. A system process called “Mergedog” periodically checks the column store tables which are loaded locally to the storage server and for which auto merge is enabled. It then determines for each individual table (or single partition of a split storage) if a merge should be executed based on configurable criteria. The configuration of the criteria is discussed in section “5.5 Configuration”.

Caution

Auto Merge is the default setting and it is recommended not to change the settings for that as this is and will be the SAP optimized way for merging tables in HANA.

You might consider turning OFF Auto Merge when loading large tables and turning it ON again once tables are loaded.

**Note**

Auto merge is used in SAP BW powered by SAP HANA's data store object activation process; see "7. SAP BW Specifics" section for more details.

5.3.2 Smart Merge

If an application requires more direct control over the merge process, SAP HANA supports a function that enables the application to request a delta merge via SQL command. The HANA system then determines if a delta merge should be executed by evaluating the delta merge decision function from in indexserver.ini. For example, if an application loads relatively large data volumes, a delta merge during the load may have a negative impact on the load performance and on other system users. Therefore, the application can disable auto merge for the tables that are being loaded and send a smart merge database "hint" to the database as soon as the load process is complete.

SQL Syntax:

```
MERGE DELTA OF <table_name> WITH PARAMETERS ('SMART_MERGE'='ON' | 'OFF')
```

As soon as the application, e.g. SAP BW, issues a smart merge "hint" to the database in order to trigger the merge processes, the following occurs:

- The system parameters configured for the "smart merge" process are evaluated
 - If the parameters indicate a merge is required, the merge is executed
 - If the "smart merge" parameters indicate a merge is not required, there is no further action and only a subsequent new hint from the application will trigger another evaluation of the parameters.

**Caution**

For tables you want to merge with the smart merge you should disable the auto merge. Otherwise it could be that the auto merge and smart merge interfere each other

**Note**

Smart merge is particularly relevant to SAP BW developers and administrators since it requires the implementation of SAP BW triggers to initiate the delta merge for certain SAP BW loading processes. More information on this topic will be provided in the "7.1 Differences by Object Type" section.

5.3.3 Other

**Note**

The following merge types are not used in SAP BW powered by SAP HANA. They are initiated either manually, using an SQL function, or automatically and described in order to give an overview of all the merge types in the database.

5.3.3.1 Memory Merge

When issuing a merge via SQL statement, optional parameters can be passed which tell the system to do the merge only in memory. As a consequence, the changes are not yet written to the persistence layer and the delta log is not truncated. The update of the delta merge to the disk is subsequently processed via merge statement via SQL.

SQL Syntax:

```
MERGE DELTA OF 'tableName' WITH PARAMETERS ('MEMORY_MERGE' = 'ON' | 'OFF')
```

5.3.3.2 Hard Merge

A hard merge is issued via a SQL statement and tells the server to execute the delta merge in any case that means without evaluating any formula from the configuration. However, the hard merge takes the token concept into account. See “5.4 Token Concept” section for more details.

SQL Syntax:

```
MERGE DELTA OF 'tableName'
```

5.3.3.3 Forced Merge

Forced merge is similar to a hard merge. It tells the system that delta storage must be merged disregarding any other factors, like system resource availability, delta storage size, etc. A forced merge may be useful in a situation where there is a heavy system load, but a small table needs to be merged or if a missed merge of a certain table is negatively impacting system performance.

SQL Syntax:

```
MERGE DELTA OF '<table_name>' WITH PARAMETERS ('FORCED_MERGE' = 'ON'|'OFF')
```



Caution

- The forced merge statement ignores the token concept



Note

The system however checks that there is not a memory issue occurring so this merge type still is not critical to the database.

5.3.3.4 Critical Merge

The critical merge aims at keeping the system stable. In a case where auto merge has been disabled and no manual merge hints are sent to the system, the delta size could grow too large for a successful delta merge. The critical merge will automatically initiate a delta merge when a certain threshold is passed. This will therefore overrule other merge settings.



Caution

- In any version prior to SPS4, no merge will be triggered automatically if the auto merge flag is turned off!

5.4 Merge Monitor

Merge Monitor controls the merge requests for all local column tables from a host global point of view. This means in a distributed system, every index server has its own Merge Monitor. For any of these, Merge Monitor only sees the host machine it is running on. The decision when to merge a single table must take the status of all column tables into account. The system uses cost functions to decide which table to merge and when to merge. There are also cost functions used to control how many tables are merged at the same time and how many threads are used to merge a single table.

Merge Monitor blocks merge request threads if there are insufficient system resources available or if the same table is already merged by another thread. This avoids long waits/delays for other threads for inserting or just reading data.

Depending upon current system resource consumption, merge motivation, cost functions and merge criteria, Merge Monitor lets single requesting merge threads pass and releases waiting threads. Each of these dependencies is discussed in the following sections.

5.5 Token Concept

Delta merge can create a heavy load on system resources. Therefore, controls need to be applied to ensure the delta merge processes do not consume all system resources.

The control mechanism is based on the allocation of tokens to each merge process. With the exception of the forced merge, a merge process cannot start unless it has been allocated tokens.

- Each merge token represents a single CPU.
- The load balancing cost function (`load_balancing_func`) determines the actual number of available merge tokens. This function is evaluated periodically (For more details refer to “5.5.2 Merge Token Configuration” section.)
- If all merge tokens are taken, merge requests have to wait for either new merge tokens through the system resources being released or tokens to be released by completed merge requests.

5.6 Configuration

In order to review the current configuration for the delta merge parameters, open SAP HANA Studio, select the server in the navigator and open the administration tools

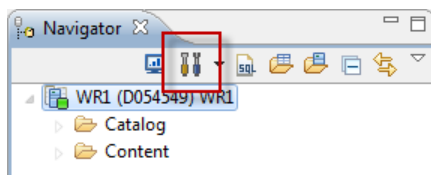


Figure: SAP HANA Navigator

Navigate to the “Configuration” tab and insert “merge” as a filter. The relevant values are displayed under the category “mergedog” and start with “smart_merge...”

Overview Landscape Alerts Performance Volumes Configuration System Information Diagnosis Files Trace Configuration	
Filter: merge	
Name	Default
attributes.ini	
[] idattribute	
check_duplicates_on_merge	history-only
indexserver.ini	
[] indexing	
merge_history_bulk_size	100000
parallel_merge_threads	2
[] mergedog	
active	yes
auto_merge_decision_func	(THM>=256000 and (((DMS>100 or DCC>100 or DLS>1000) and DRC > MRC/100) or (DMR>0.2*MRC and DMR > 1000))) or (THM<256000 and (DMS>50 or DCC>8 or DL...
auto_merge_priority_func	1 / (7 + MMS)
check_interval	60000
critical_merge_decision_func	UPT > 43200 and ((MMS<10000 and DMS>1000 and TMD>86400) or TMD>604800)
hard_merge_priority_func	QDW
load_balancing_func	1 + LCC * (AHM/THM) * (100-CLA)/100
max_delta_memsize	1000
memory_merge_decision_func	
min_cid_threshold	1000000
smart_merge_decision_func	MMS<1000 or DMS>1000 or DRC>0.1*MRC or DMR>0.1*MRC or DLS>5000
smart_merge_enabled	yes
smart_merge_priority_func	1 / (7 + MMS)
token_per_table	2

Figure: SAP HANA Configuration

The SAP BW powered by SAP HANA relevant decision settings are identified by the values starting with “smart_merge...”.



Note

```
[mergedog]
active = yes
smart_merge_enabled = yes
```

The parameters in the above configuration indicate that:

Smart merge decision function evaluates true if delta memsize (DMS) exceeds 1 GB or delta cell count (DCC) exceeds 800 Mio or delta logsize (DLS) exceeds 5 GB. The DMS (delta mem size) determines the priority.

In other words, if the delta table memory size is greater than 1000 MB or the cell count in the delta table is greater than 800 million or the delta log size is greater than 5000 MB the delta merge will be executed by smart merge if a smartmerge HINT is received by the Merge Monitor.

The system prioritizes smart merge requests based on the delta memory size (DMS). Tables with the bigger deltas are merged first.

Smart merge can be switched off completely by changing the following parameter:

```
[mergedog]
enable_smart_merge = no
```



Caution

- Mergedog settings should not be changed unless instructed to do so by SAP Support

5.6.1 Cost Function

The SAP HANA database will decide whether a requested delta merge should be executed or not, based on a cost function calculated during runtime. The cost function that determines if the requested delta merge will be executed is configured in the parameter

smart_merge_decision_func. The cost function that determines which priority will be assigned to the delta merge is the smart_merge_priority_func.



Note

The evaluation of the cost function will only happen once for each smart merge request. So if the cost function yields a result of false (i.e. the system decides that a delta merge is not required), this request (hint) will be logged and the request will not be evaluated again. Only a new request (hint) can potentially initiate a new delta merge.

SAP HANA in the current version of writing this “How To” guide provides the following parameters to configure the cost function:

Derived From an Actual Table:

- DMS delta memory size [MB], is zero MB for an unloaded table
- MMS main resident memsize [MB], this is zero MB for an unloaded table
- TMD table merge delay [sec]
- MRC main row count [million]
- DMR deleted main rows [million]
- DLS delta log size [MB]
- DCC delta cell count [million]
- DRC delta row count [million]
- DUC delta uncommitted row count [million]
- NAME table name [string]
- SCHEMA schema name [string]
- LOADED table delta currently loaded [boolean]
- RHM estimated required heap memory to complete table optimization [MB]

Derived From an Optimization Request:

- QDW queuing delay wait [sec]

Independent From Table and Request:

- CLA CPU load average [percentage]
- LCC logical CPU count
- THM total heap memory [MB]
- AHM available heap memory, including memory which could be freed [MB]
- UPT indexserver uptime [sec]

Other Options:

- NOW() current date and time (localtime of the server timezone) as date

SAP HANA SPS09 and Higher:

- NMU main max udiv [million]
- OCRC (last) optimize compression row count [million]

- CRCSOC change row count since (last) optimized compression [million]
- RP table is range partitioned [Boolean]

Parameters derived from merge requests can be used to build cost functions related to merge requests only, like all table merge request priority functions. For decision cost functions, optimization request related parameters can't be used. Non-table related parameters like CPU load average or memory consumption can be used for almost all cost functions. For decision cost functions, all parameters may be used except those derived from optimization request. This is obvious however as the decision to merge or optimize a table must be made prior to the actual merge request starting.



Note

- All these parameters can be used to build cost functions for all delta merge configurations.

5.6.2 Merge Token Configuration

For every merge request, the number of tokens required to perform the merge is calculated by the system. If the system is not able to determine a value, a default value will be returned. This default value can be configured through the token_per_table parameter. (It is not recommended to change this value

[] mergedog	
active	yes
auto_merge_decision_func	DMS>1000 or TMD>3601 or DCC>800 or DL...
auto_merge_priority_func	DMS/1000 + TMD/3601 + DCC/800
check_interval	60000
hard_merge_priority_func	QDW
load_balancing_func	1 + LCC * (AHM/THM) * (100-CLA)/100
max_delta_memsize	1000
memory_merge_decision_func	
min_cid_threshold	1000000
smart_merge_decision_func	DMS>1000 or DCC>800 or DLS>5000
smart_merge_enabled	yes
smart_merge_priority_func	DMS/1000
token_per_table	2

Figure: Merge Token Configuration 1

The number of merge tokens available for allocation through a process will be adjusted following the current system resource availability. This number will be recalculated periodically by the system based on a cost function (load_balancing_func). The default cost function formula:

[] mergedog	
active	yes
auto_merge_decision_func	DMS>1000 or TMD>3601 or DCC>800 or DLS>5000
auto_merge_priority_func	DMS/1000 + TMD/3601 + DCC/800
check_interval	60000
hard_merge_priority_func	QDW
load_balancing_func	1 + LCC * (AHM/THM) * (100-CLA)/100

Figure: load_balancing_func Default Configuration



Note

- If a hard maximum is required for the amount of tokens available, a constant value can be configured or a constant parameter (e.g.: LCC) could be used.

```
load_balancing_func = 1 + LCC * (100-CLA)/100
```

There is no option or need to disable, switch off, stop or even kill MergeMonitor. MergeMonitor is not a thread.

5.7 General SQL Syntax for Delta Merge

SQL Syntax:

```
MERGE [HISTORY] DELTA OF table_name [PART int_const] [ WITH PARAMETERS '('  
parameter_key_value_list ')'] ]
```

WITH PARAMETERS (parameter_key_value):

Column store-specific options can be passed in using the "WITH PARAMETERS" clause.

- Keys and single values can be any string constant
- Duplicate keys are allowed
- Keys are automatically mapped into their uppercase representation

parameter_key_value ::=

<string_literal> = <string_literal> |

<identifier> = <string_literal> |

<string_literal> = (<string_literal>, ...

Current parameters

'SMART_MERGE' = 'ON' | 'OFF'

When SM ART_MERGE is ON, the database does a smart merge, this means database decides whether to merge or not based on merge criteria specified in automerge section of indexserver configuration

'PASSPORT'='<string>'

Merge request call tag to identify the request in related traces

'MEMORY_MERGE'='ON'|'OFF'

Database merges delta index in memory only, it will not be persisted.

'FORCED_MERGE'='ON'|'OFF'

If you want the merge to take place immediately regardless of system resource availability

Description

The **MERGE DELTA** statement merges the deltas to main column store table. Since the column store is read optimized and compressed, delta tables are introduced to optimize insert or updates in the optimized way. All insertions are passed to a delta index. At a certain point in time, deltas can be merged into the main index. Deltas will be merged into main column table and history deltas will be merged into history main. Regarding delta merge, historization of data is done within delta merge phase (of normal delta, not history table).

6. Monitoring

6.1 Merge History

All smart merges are logged and are found in the monitoring view M_DELTA_MERGE_STATISTICS. There you can find several delta merge details like table name, merge start time, merge duration and so on.

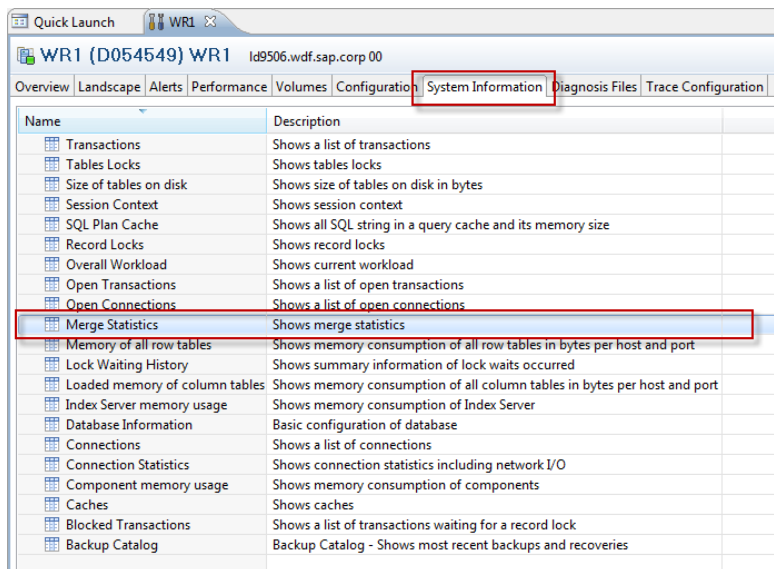


Figure: System Information

1000 rows (3213 ms) Max rows: 1000 Refresh

Raw Data Distinct values Enter your filter Filtered rows: 1000/1000 Add filter Save as file

ID	HOST	PORT	TYPE	SCHEMA_NAME	TABLE_NAME	PART_ID	HISTORY	MEMORY_MERGE	PASSPORT	START_TIME	EXECUTION_TIME	MOTIVATION	SUCCESS
H5006	30.103	MERGE	SAPCIA	/BIC/AURDSO4R80	0	FALSE	FALSE			26.03.2012 10:36:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AZHSSMART80	0	FALSE	FALSE			26.03.2012 10:36:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AZINV_D0180	0	FALSE	FALSE			26.03.2012 10:36:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AZINV_D0280	0	FALSE	FALSE			26.03.2012 10:36:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AZNVW_D0180	0	FALSE	FALSE			26.03.2012 10:36:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AZNVW_D0180	0	FALSE	FALSE			26.03.2012 10:36:24	0	AUTO	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0100	0	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:36	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0100	0	TRUE	FALSE	BW merge trigge...		26.03.2012 10:36:36	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0140	0	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:36	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0200	0	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:44	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0200	0	TRUE	FALSE	BW merge trigge...		26.03.2012 10:36:44	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/APNVW_D0240	0	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:44	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	1	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:52	1	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	2	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:52	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	3	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:52	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	4	FALSE	FALSE	BW merge trigge...		26.03.2012 10:36:52	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	1	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	2	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	3	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C01	4	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C02	1	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C02	2	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C02	3	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	HINT	SAPCIA	/BIC/FPNVW_C02	4	FALSE	FALSE	BW merge trigge...		26.03.2012 10:37:13	0	SMART	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AHADSO180	0	FALSE	FALSE			26.03.2012 10:37:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AHADSO280	0	FALSE	FALSE			26.03.2012 10:37:24	2	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AHANA_DS080	0	FALSE	FALSE			26.03.2012 10:37:24	3	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AHSD_001C80	0	FALSE	FALSE			26.03.2012 10:37:24	1	AUTO	FALSE
H5006	30.103	MERGE	SAPCIA	/BIC/AHSD_001D80	0	FALSE	FALSE			26.03.2012 10:37:24	1	AUTO	FALSE

Figure: Merge Statistics

All smart merge calls by the application are listed. For those the "TYPE" column equals "HINT". Column "SUCCESS" indicates the decision whether a smart merge actually was triggered or not. The application hint was accepted and a table delta merge request was started asynchronously.

All dropped application smart merge hints are listed with type "HINT" and success "FALSE". There are no entries of type "MERGE" for dropped application hints for smart merge.

The reason why smart merge hint was dropped will be explained with column "LAST_ERROR":

Last Error	Description
2481	same table optimization requested multiple times; there were already some other smart merge calls waiting for same table
2482	table optimization was not indicated; smart merge cost function evaluation result was false
2483	table optimization was disabled; smart merge feature was switched off by storage server configuration
2484	not enough memory for table optimization, table optimization memory requirement exceeds heap limit
2480	same table optimization is ongoing; table already gets optimized by another thread, only if automerger

Table: Last Error Code Description



Note

- Even if the entry of type "HINT" has the status "SUCCESS", it doesn't necessarily mean that the delta merge was successful. The actual merge still might fail.

For all accepted application smart merge requests, there was an actual delta merge done by the system. For this delta merge you find a second entry with "TYPE" column as "MERGE".

For the optimization of the compression algorithm mentioned in Step 2 of "5.1 Merge Process" section, you find a third entry with type SPARSE. This follows a successful delta merge.

RB	HOST	12	PORT	RB	TYPE	RB	SCHEMA_NAME	RB	TABLE_NAME	12	PART_ID	RB	HISTORY	RB	MEMORY_MERGE
It5006	30.103				SPARSE		SAPCIA		DB6PMPROT	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		SAPWLSERV	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		RSABWN_USR_TREE	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		SWNCMONIINDEX	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		SWNCCOLLPERF	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		RSMDATASTATE_D...	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		RSBKDTPTH	0			FALSE		FALSE
It5006	30.103				SPARSE		SAPCIA		RSBKDTPTH	0			FALSE		FALSE

Figure: Merge Statistics With Type SPARSE

For smart merges check column with name "MOTIVATION" marked as "SMART". Column "PASSPORT" identifies individual smart merge requests. Column "SUCCESS" indicates whether the table delta actually was merged.

The delta merge history lists all delta merges like smart merge, hard merge, auto merge and critical merge. Besides completed delta merges, the history also contains all application merge hints.

The table delta merge history consists of two parts:

- The storageserver delta merge statistics.

This contains all delta merges and application hints since last system restart. This is a short term history. The number of entries is limited to the latest 1 Mio. It will be updated as soon as the merge was completed or an application hint was received.

- The statisticsserver delta merge statistics.

It contains delta merges of the past, independent of system restarts. This is a long term history. It will be updated periodically. Generally it does not contain the latest delta merges.

In case of a missed smart merge of table 'xyz' work thru the following:

1) Look for table smart merges in storageserver merge statistics.

```
select * from M_DELTA_MERGE_STATISTICS where table_name = 'xyz' and motivation = 'SMART'
```

If you cannot find any:

2) Look for application merge hints in storageserver merge statistics:

```
select * from M_DELTA_MERGE_STATISTICS where type = 'HINT' and table_name = 'xyz'
```

If there was no hint from the application, the system won't initiate a delta merge.



Note

- If there was a hint, the system only smart merges if the criteria for smart merge are fulfilled. The result is listed in the statistics. The smart merge criteria are defined as smart merge decision cost functions as part of the storage server configuration.

If you do not find anything related to the table:

3) Visit the long term statisticsserver table delta merge history:

```
select * from _SYS_STATISTICS.HOST_DELTA_MERGE_STATISTICS where table_name = 'xyz'
```

6.2 Alerts

There are certain alerts in SAP HANA related to the delta merge (e.g.: delta memory size, etc.).



Note

The alerts are only visible via the SAP HANA studio. There are no SAP BW specific logs which can be viewed from within SAP BW.

Overview Landscape Alerts Performance Volumes Configuration System Information Diagnosis Files Trace Configuration

Last check: 26.03.2012 10:11:27 Show: current alerts only Filter: delta

Alert filter: Alert contains 'del...' All default ratings are selected

Time	Description	Priority	Host
Current Alerts (1/6)			
26.03.2012 09:1	SAPCIA./BIC/FPZIMC4N has 28784322 records in delta memory	MEDIUM	*
26.03.2012 09:1	SAPCIA./BIC/FZMHXP01 has 1972800 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AHANA_DSO00 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AHANA_SM100 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AHANA_SM300 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AHANA_SMP00 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AHSD_O01B40 has 28789644 records in delta memory	MEDIUM	*
26.03.2012 09:1	SAPCIA./BIC/AURDSO4N00 has 4927570 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/AURDSO4P00 has 28784322 records in delta memory	MEDIUM	*
26.03.2012 09:1	SAPCIA./BIC/AXTE11_SH00 has 10439612 records in delta memory	MEDIUM	*
26.03.2012 09:1	SAPCIA./BIC/AZINV_D0100 has 2630400 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0003603000 has 1733277 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0003604000 has 9288828 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0003613000 has 1591704 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0003614000 has 4092905 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0009516000 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0009517000 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0009518000 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/B0009791002 has 8764800 records in delta memory	LOW	*
26.03.2012 09:1	SAPCIA./BIC/FHANA_0001 has 1792762 records in delta memory	LOW	*
26.03.2012 09:1	Delta memory 1571 MB exceeds max delta memory size 1000 MB for SAPCIA./BIC/FPZIMC4...	LOW	*
26.03.2012 09:1	Delta memory 2755 MB exceeds max delta memory size 1000 MB for SAPCIA./BIC/AHSD_O0...	MEDIUM	*
26.03.2012 09:1	Delta memory 1389 MB exceeds max delta memory size 1000 MB for SAPCIA./BIC/AURDSO...	LOW	*
26.03.2012 09:1	Delta memory 8588 MB exceeds max delta memory size 1000 MB for SAPCIA./BIC/AURDSO...	HIGH	*
26.03.2012 09:1	Delta memory 3189 MB exceeds max delta memory size 1000 MB for SAPCIA./BIC/AXTE11_S...	MEDIUM	*

Figure: Alerts in SAP HANA

The Figures “Configuration of Warnings Based on Delta Memory Size” and “Configuring Delta Memory Thresholds” below are displaying some examples of the configuration of alert thresholds which can be found in the statisticserver.ini file.


Overview	Landscape	Alerts	Performance	Volumes	Configuration	System Information	Diagnosis Files	Trace Configuration
Filter: <input type="text" value="merge"/> 								
Name		Default						
- statisticsserver.ini <ul style="list-style-type: none">- [] mergedog<ul style="list-style-type: none">- [] statisticsserver_array_COLUMN_TABLES_LAST_MERGE_TIME- [] statisticsserver_array_DELTA_MERGE_EXECUTION_TIME- [] statisticsserver_array_DELTA_MERGE_HOST- [] statisticsserver_array_DELTA_MERGE_MEMORY_MERGE- [] statisticsserver_array_DELTA_MERGE_MERGED_DELTA_RECORDS- [] statisticsserver_array_DELTA_MERGE_MOTIVATION- [] statisticsserver_array_DELTA_MERGE_PART_ID- [] statisticsserver_array_DELTA_MERGE_PASSPORT- [] statisticsserver_array_DELTA_MERGE_PORT- [] statisticsserver_array_DELTA_MERGE_SCHEMA_NAME- [] statisticsserver_array_DELTA_MERGE_START_TIME- [] statisticsserver_array_DELTA_MERGE_SUCCESS- [] statisticsserver_array_DELTA_MERGE_TABLE_NAME- [] statisticsserver_array_DELTA_MERGE_TYPE- [] statisticsserver_monitor_DELTA_MEM_MERGE_DOG<ul style="list-style-type: none">- description<ul style="list-style-type: none">- Check size of delta memory and merge dog configuration- foreach<ul style="list-style-type: none">- COLUMN_TABLES_PART_SIZE_SCHEMA_NAME- id<ul style="list-style-type: none">- 29- intervals<ul style="list-style-type: none">- interval_1h- label<ul style="list-style-type: none">- "Delta memory" + COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 + " MB exceeds max delta memory size" + MERGE_DOG_MAX_DELTA_MEM_SIZE + " MB for- name<ul style="list-style-type: none">- Check size of delta memory- scope<ul style="list-style-type: none">- system- useraction<ul style="list-style-type: none">- Check if mergedog works- warning1<ul style="list-style-type: none">- COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA_MEM_SIZE*THRESHOLD_DELTA_MEM_MERGE_DOG_WARNING_LEVEL_1- warning2<ul style="list-style-type: none">- COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA_MEM_SIZE*THRESHOLD_DELTA_MEM_MERGE_DOG_WARNING_LEVEL_2- warning3<ul style="list-style-type: none">- COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA_MEM_SIZE*THRESHOLD_DELTA_MEM_MERGE_DOG_WARNING_LEVEL_3- [] statisticsserver_monitor_MON_PARAM_MEMWATCH_DELTA_MERGE- [] statisticsserver_parameter_MERGE_DOG_MAX_DELTA_MEM_SIZE- [] statisticsserver_parameter_PARAM_MEMWATCH_DELTA_MERGE								

Figure: Configuration of Warnings Based on Delta Memory Size


Overview Landscape Alerts Performance Volumes Configuration System Information Diagnosis Files Trace Configuration	
Filter: <input type="text" value="merge"/> 	
Name	Default
name	Check size of delta memory
scope	system
useraction	Check if mergedog works
warning1	COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA
warning2	COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA
warning3	COLUMN_TABLES_PART_SIZE_MEMORY_SIZE_IN_DELTA[]/1024/1024 > MERGE_DOG_MAX_DELTA
▶ [] statisticserver_monitor_MON_PARAM_MEMWATCH_DELTA_MERGE	
▶ [] statisticserver_parameter_MERGE_DOG_MAX_DELTA_MEM_SIZE	
▶ [] statisticserver_parameter_PARAM_MEMWATCH_DELTA_MERGE	
▲ [] statisticserver_parameter_THRESHOLD_DELTA_MEM_MERGE_DOG_V	
description	Factor of delta memory to max_delta_memsize of mergedog. Threshold for warning level 1.
formula	1
intervals	interval_1h
scope	system
▲ [] statisticserver_parameter_THRESHOLD_DELTA_MEM_MERGE_DOG_V	
description	Factor of delta memory to max_delta_memsize of mergedog. Threshold for warning level 1=>2.
formula	2
intervals	interval_1h
scope	system
▲ [] statisticserver_parameter_THRESHOLD_DELTA_MEM_MERGE_DOG_V	
description	Factor of delta memory to max_delta_memsize of mergedog. Threshold for warning level 1=>3.
formula	5
intervals	interval_1h
scope	system

Figure: Configuring Delta Memory Thresholds

7. SAP BW Specifics

7.1 Differences by Object Type

The merge process operation differs for data loading into different SAP BW objects. The delta merge is either performed automatically by the application or must be triggered manually within the application. This depends on the relevant object type:

Object Type	Delta Merge Trigger
Persistent Staging Area (PSA)	The delta merge is automatically performed after the system writes to the PSA using the smartmerge
Master Data	The system uses automerge
Standard/SAP HANA Optimized DataStore Object	The delta merge is automatically performed in the activation process. (This also applies to DataStore objects that belong to a semantically partitioned object.)
Advanced DataStore Object	For DataStore objects (advanced), the Trigger Database Merge checkbox is selected by default. After activation, an automatic check is run in order to ascertain whether a delta merge can be performed.
Write-Optimized DataStore Object	<p>The delta merge is not performed automatically. (This also applies to objects that belong to a semantically partitioned object.)</p> <ul style="list-style-type: none"> • The Data Transfer Process (DTP) has an Update tab that contains the Trigger Database Merge checkbox. This controls the delta merge, once the DTP request has been successfully processed. By default, this checkbox is selected. (see Figure “DTP Configuration”) • There is a Process Type, NewDB Merge, that can be used within a Process Chain to execute the “smartmerge”
Standard/SAP HANA Optimized InfoCube	<p>The delta merge is not performed automatically. (This also applies to objects that belong to a semantically partitioned object.)</p> <ul style="list-style-type: none"> • The DTP has an Update tab that contains the Trigger Database Merge checkbox. This controls the delta merge, once the DTP request has been successfully processed. By default, this checkbox is selected. (see Figure “DTP Configuration”) • There is a Process Type, NewDB Merge, that can be used within a Process Chain to execute the “smartmerge”

Table: SAP BW Object Types



Caution

Keep in mind the hard limit of 2 billion rows per partition also applies to the delta storage. Additionally, as stated in section “5.1 Merge Process,” delta merge requires a certain amount of additional memory which might cause an out of memory problem if the delta storage is not merged from time to time and gets too large.

7.1.1 DTP Trigger Setting for Invoking “Smartmerge” Hint

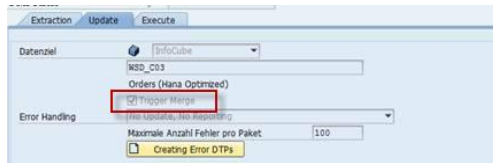


Figure: DTP Configuration

Note

In certain exceptional cases, performing a delta merge after processing a DTP request is not recommended, due to load balancing issues. In these cases only, we recommend that you deselect the checkbox in the DTP and use the Trigger for “NewDB” Merge Process Type to trigger the merge. For example, data from multiple sources is loaded into an object. The merge for this object should only be performed at the end of the entire loading process.

7.1.2 Process Type to Trigger “Smartmerge” Hint

If you use the Trigger Delta Merge process type, proceed as follows: In the plan view of the process chain, the Process Type Trigger Delta Merge is available in the Load Process and Post-Processing process category:

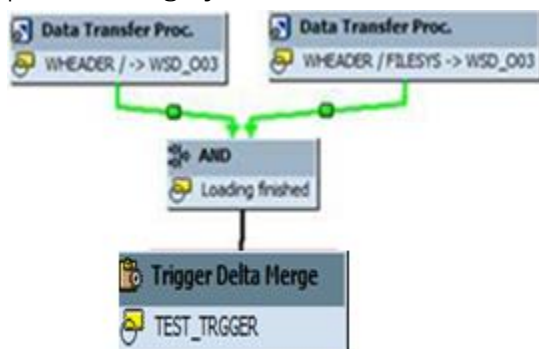




Figure: Process Chain

1. Drag and drop the Trigger Delta Merge process type into the process chain.
2. A dialog box appears. To create a new process variant, choose  (Create).
3. In the next screen, enter a name and a description for the process variant and choose  (Continue).
4. On the process variant maintenance screen, specify the type and name of the object for the delta merge.
5. Save your entries and return to the plan view of the process chain.
6. Link the process to the required loading processes.

For more information, see Creating Process Chains.

Note

- Make sure a delta merge is triggered by either the DTP or the process type. Otherwise, the data remains in the delta storage table and over time this leads to suboptimal memory use and read performance. It can also lead to data loss or corruption.



Note

As stated in section “5.5 Configuration,” there are certain parameters that define if the merge is actually executed in SAP HANA when the trigger was received.

7.1.3 BW Application Log entries for Delta Merge Trigger

If a smartmerge HINT is triggered from BW, an entry is created in the application log. Use transaction SLG1 to view the application logs.

2012-04-02 14:31:21	ALEREMOTE	2	RSDS 0TCTBWOBCT_TEXT	CIACLT003	RSHDB	MERGE	CL_RSO_APPLICATION_LOG=====CP	Dialog processing
2012-04-02 16:49:19	QT6CLNT004	2	RSDS 2LIS_11_VAHDR	QT6CLNT004	RSHDB	MERGE	CL_RSO_APPLICATION_LOG=====CP	Dialog processing
Problem class Additional information								
		2						

Message Text

BW merge trigger for RSDS 2LIS_11_VAHDR QT6CLNT004 63

NewDB merge called for /BIC/B0009777000

Figure: Process Chain Log Entry for Smartmerge HINT triggered



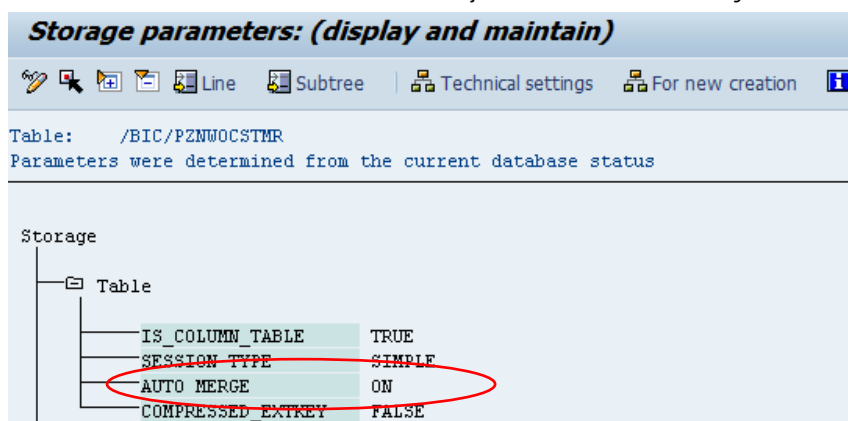
Note

See SAP Note 1663501

7.2 Review Automerger Setting by Table

The automerger setting assigned to any BW table can be viewed in the SAP BW application via the following path:

Tcode SE11 -> Utilities -> Database Object -> Database Utility -> Storage Parameters (button)



Storage parameters: (display and maintain)

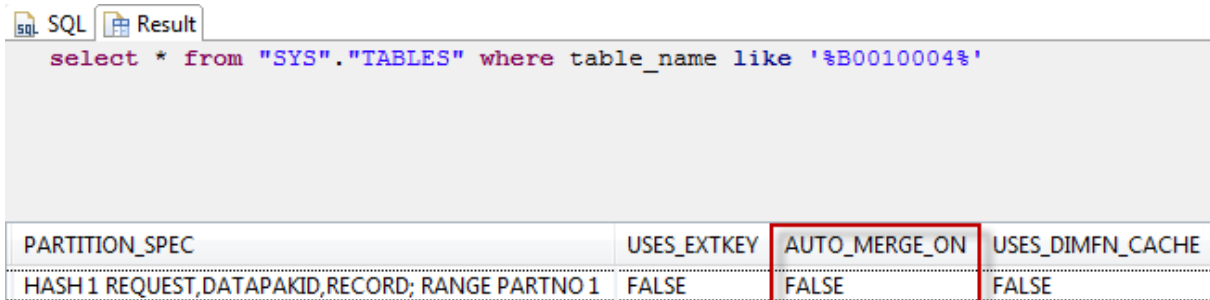
Table: /BIC/PZNWOCSTMR
Parameters were determined from the current database status

Storage	
Table	
IS_COLUMN_TABLE	TRUE
SESSION TYPE	SIMPLE
AUTO MERGE	ON
COMPRESSED_EXTKEY	FALSE

Figure: Storage Parameters in SAP BW

The setting can also be reviewed from the SAP HANA Studio with the following SQL:

```
select * from "SYS"."TABLES" where table_name like '%<table name>%'
```

The screenshot shows the SAP HANA Studio interface. At the top, there are two tabs: 'SQL' and 'Result'. The 'SQL' tab is active, displaying the following query:

```
select * from "SYS"."TABLES" where table_name like '%B0010004%'
```

Below the query, the 'Result' tab is active, showing a table with the following data:

PARTITION_SPEC	USES_EXTKEY	AUTO_MERGE_ON	USES_DIMFN_CACHE
HASH 1 REQUEST,DATAPAKID,RECORD; RANGE PARTNO 1	FALSE	FALSE	FALSE

The 'AUTO_MERGE_ON' column is highlighted with a red border.

Figure: Review Table Settings in SAP HANA Studio

**Caution**

These settings are system generated. SAP will not support any changes made to the table settings.

8. Appendix

8.1 Appendix A - Temporal Tables for Data Store Object Change Log

The SAP HANA database supports temporal tables that allow queries against a historical state of the data. Write operations on temporal tables do not physically overwrite existing records. Instead, write operations always insert new versions of the data record into the database. The different versions of a data record have timestamp-like attributes that indicate their validity.

The most recent versions in temporal tables are called current data. All other versions of the same data object contain historical data. Currently, SAP BW powered by SAP HANA does not support time-based queries on historical versions.

Temporal tables are used in SAP BW powered by SAP HANA for the active data tables in order to maintain the change log.



Note

Note: Temporal tables must not be confused with temporary tables, which are deleted when the database session is over.

8.2 Appendix B - Cost Function Examples

Per default, if no actual priority cost function is defined, the Merge Monitor assigns the delta merge priority according to standard default cost functions.

- In the case of automerge the default function calculates priority according to the delta table memory size, cell count and merge delay:
e.g. $auto_merge_priority_func = DMS/1000 + TMD/3601 + DCC/800$
- The smartmerge priority function has no time component:
e.g. $smart_merge_priority_func = DMS/1000$



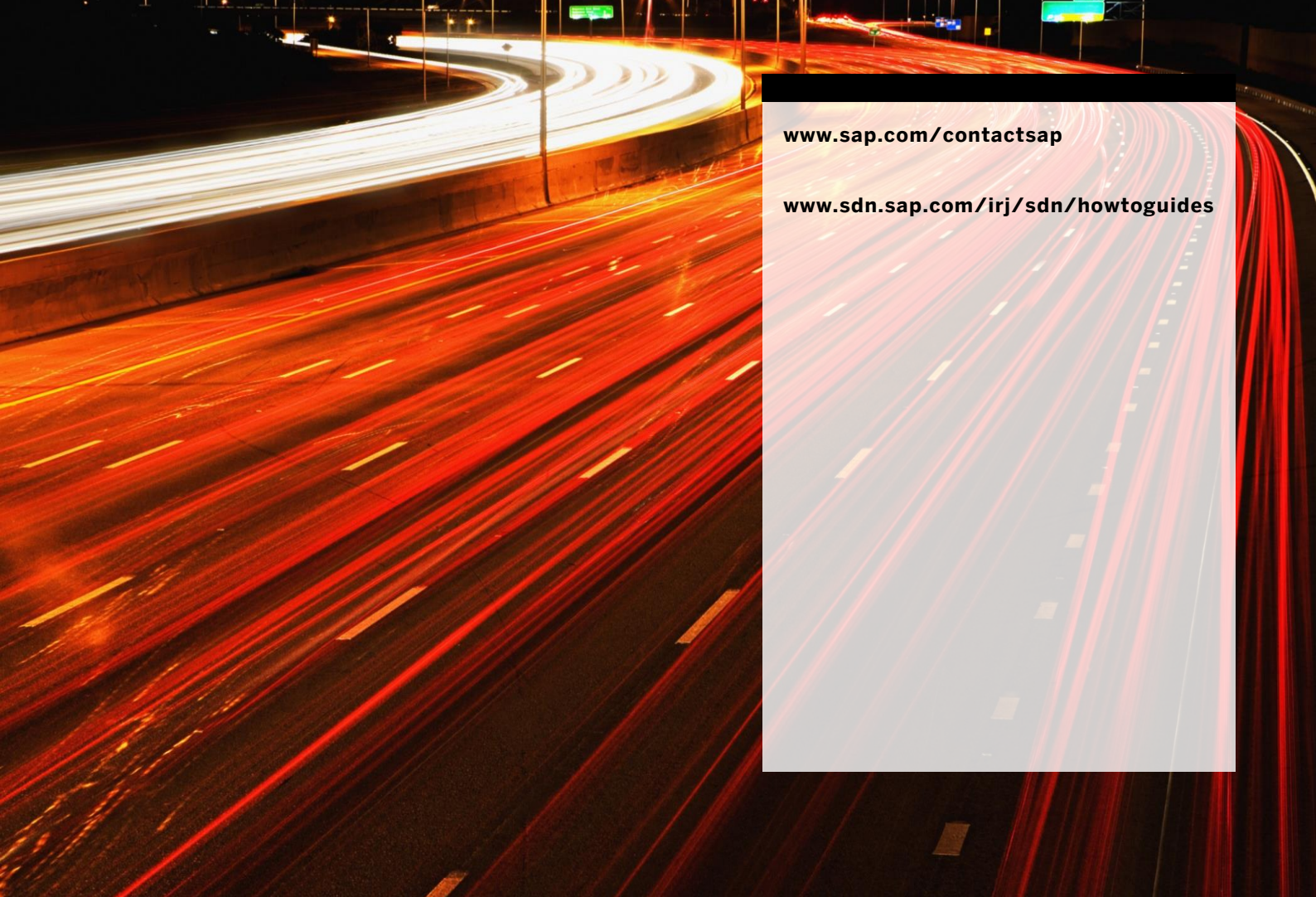
Note

- With the default configuration of the smartmerge priority cost functions, the largest delta table in memory eligible for merge is merged first.
- Automerge priority cost function takes merge delay into account.
- SmartMerge comes first. All Automerges got lower priority than any smartmerge.

More examples for cost functions are listed below:

```
smart_merge_decision_func = leftstr(NAME,1)='Z' and DMS > 1000 and TMD >= 86400
```

```
smart_merge_decision_func = (CLA < 30 and DMS > 1000) or DMS > 5000
```

www.sap.com/contactsap

www.sdn.sap.com/irj/sdn/howtoguides



The Best-Run Businesses Run SAP™