Optimization of Memory: SAP HANA Database Perspective - PART 2

Rajesh Azmeera[1], Rhea Khanna[2], Deepak Nanuru Yagamurthy[3]

[1]Technology Professional, Department of Information and Technology, Stryker, US (Corresponding Author)

[2]Technology Professional, Department of Research, Microsoft, US

[3]Technology Professional, Department of Information and Technology, 7 Leven Inc, US

Abstract: ***Memory is a fundamental resource of the SAP HANA database. Understanding how the SAP HANA database requests, uses, and manages this resource is crucial to the understanding of SAP HANA. As Data is growing rapidly and these data is stored in memory to process. Increasing DB size is directly proportional to its memory consumed. SAP HANA provides a variety of memory usage indicators that allow for monitoring, tracking, and alerting. The most important indicators are used memory and peak used memory. Memory needs to be optimized, otherwise increasing Data size can't be accommodated by existing memory. Since SAP HANA contains its own memory manager and memory pool, external indicators such as the size of resident memory at host level and the size of virtual and resident memory at process level can be misleading when you are estimating the real memory requirements of an SAP HANA deployment. SAP HANA integrates data from multiple areas within an organization, for example: Traditional business documents – including contracts and spreadsheets. UX/UI (User Experience/User Interface) – including website forms, emails and other customer interactions Mobile – information from the mobile devices of customers and your workforce. IoT (Internet of Things) – data from the many sensors that run in every aspect of a business, from warehouses and trucks to stores and offices. The SAP HANA system not only integrates all of this data; it can also apply machine learning and AI to analyze it instantly and deeply, accelerating real-time decision-making by providing key insights into a company's operations.***

Keywords: SAP Memory Management, SAP HANA, Memory Optimization, Resource Utilization and SAP HANA Database

1. Introduction

SAP HANA (High-performance Analytic Appliance) is a multi-model database that stores data in its memory instead of keeping it on a disk. The column-oriented in-memory database design allows you to run advanced analytics alongside high-speed transactions – in a single system. Why is this so important? Because it lets companies process massive amounts of data with near-zero latency, query data in an instant, and become truly data-driven. By storing data in column-based tables in main memory and bringing online analytical processing (OLAP) and online transactional processing (OLTP) together, SAP HANA is unique – and significantly faster than other database management systems (DBMS) on the market today.

Pre requisite: To understand this paper thoroughly pre requisite is Optimization of Memory: SAP HANA Database perspective – PART 1

2. Discussion

Delta Storage Optimization

Table partitioning allows you to optimize the size of tables in memory and their memory consumption as each partition has its own delta storage.

The memory consumption of a table in memory during a merge operation depends on the number of records, the number and memory size of columns and the memory size of the table. While the number of records can be kept low by triggering a smart merge from the application, optimization with regards to the size of the table can be achieved by table partitioning. This is due to the fact that each partition holds a separate delta storage. When a merge is performed, the data from the main storage has to be loaded into memory which is a considerably less amount when only a single partition is handled rather than the full table.

\When considering partitioning it is recommended to analyze the typical usage of this table. Partitions should be created in a way that avoids as much as possible that single statements need to access multiple partitions. If no application knowledge is available, then hash partitioning with a partition size of about 500.000.000 records is a good initial choice

3. Case Study

System Locked Due to Missing, Expired, or Invalid License

New installations of SAP HANA are equipped with a temporary license that expires after 90 days. To keep the system functional after this period, you have to install a permanent license. Improper licensing may lead to a lockdown of your SAP HANA system. In this case, the only allowed action is to install a valid license. The system goes into lockdown in the following situations:

The first temporary license of 90 days has expired. The permanent license has expired and it was not renew within 28 days. An old backup was used for recovery and the license key in the backup has expired in the meantime

The installed license key is an enforced license key and current memory consumption exceeds the amount specified in the license key. All license keys installed in the database have been deleted

Identify and Assess Blocked Transaction Issues

The first sign of blocked transactions is poor application response or alerts 49 (Long-running blocking situations) or 59 (Percentage of transactions blocked) are raised. To confirm the database performance is harmed by blocked transactions, you should check the following SAP HANA studio monitors under the Performance tab:

Job Progress Monitor

It shows currently running SAP HANA background processes like Delta Table Merge. Since the Delta Table Merge needs to lock tables to proceed, it is a common cause for blocked transactions. For deeper analyses check chapter Job Progress Monitoring and SAP HANA Troubleshooting and Performance Guide for Save point Performance.

Session Monitor

The Session Monitor lists all currently opened SQL sessions (meaning user connections). Check chapter Session Monitoring. Blocked Transaction Monitor Lists those transactions that are currently blocked. The ordering is done via a blocking/blocked relation. Check chapter Blocked Transaction Monitoring. Thread Monitor: The Thread Monitor allows the most fine-grained view into the current situation by listing all threads in the system. Check chapter Thread Monitoring.

Note that in case that problem for scale-out configuration is visible only on one node, there is high possibility that blocking situation is caused by physical resource like network or disk.

Troubleshooting Blocked Transactions

When troubleshooting blocked transactions, it is helpful to differentiate between situations where only single or a few transactions are blocked from the situation where a high percentage of all transactions are blocked. Single or Few Transactions are Blocked. If you identified only a single or a few blocking transactions, there is likely an issue on application side. A usual pattern is a flaw in the application coding that does not commit a write transaction. Such a transaction will be a blocker for any other transaction that needs to access the same database object. To release the situation you have to close the blocking transaction.

Many Transactions are Blocked

In the case that a large amount of transactions are blocked, check chapter Analyses of blocking transactions.

Troubleshooting Blocked Transaction Issues that Occurred in the Past

Tools such as the Load Monitor, system views and the SQL Plan cache are available for finding the root cause of blocked transaction situations that were resolved.

Load Monitor isolates the exact time frame where the issue happened. Using that information, investigate what happened at this specific time frame. You should check the following monitoring and Statistic Server views: **SYS_STATISTICS.HOST_BLOCKED_TRANSACTIONS**: Analyze the columns “WAITING_SCHEMA_NAME”, “WAITING_TABLE_NAME” and “WAITING_RECORD_ID” to identify the database objects that lead to blocked transactions

SYS.M_DELTA_MERGE_STATISTICS: The column “START_TIME” and “EXECUTION_TIME” provide you with the information if there was a Delta Table Merge running A longer history can be found in the Statistic Server table _SYS_STATISTICS.HOST_DELTA_MERGE_STATISTICS

SYS.SAVEPOINTS: Check if a save point was written during the time period. A longer history can be found in _SYS_STATISTICS.HOST_SAVEPOINTS

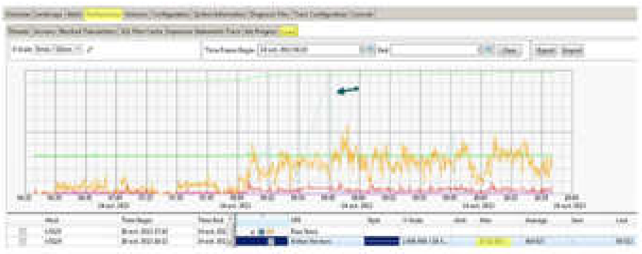
SQL Plan Cache monitor may be able to provide information about the statements that were involved in the situation.

Multiversion Concurrency Control (MVCC) Issues

Multiversion Concurrency Control (MVCC) is a concept that ensures transactional data consistency by isolating transactions that are accessing the same data at the same time.

To do so, multiple versions of a record are kept in parallel. Issues with MVCC are usually caused by a high number of active versions. Old versions of data records are no longer needed if they are no longer part of a snapshot that can be seen by any running transaction. These versions are obsolete and need to be removed from time to time to free up memory. This process is called Garbage Collection (GC) or Version Consolidation. It can happen that a transaction is blocking the garbage collection. The consequence is a high number of active versions and that can lead to system slow-down or out of memory issues. Problems with high number of active versions can be identified by: Users report an increase of response times

The index server trace contains that there are too many un-collected versions. The transaction blocks the garbage collection of HANA database. By checking “Active Versions” in the Load Monitor (see Performance tab)



Transactions blocking garbage collection can originate from:

Long-running or unclosed cursors. Long-running transactions with isolation mode “serializable” or ”repeatable read”

Hanging threads

In order to validate there is a problem with MVCC, check the number of active versions in the Row StoreMVCC manager monitoring view:mselect * from m_mvcc_tables where host='<host>' and port='<port>' and (name='NUM_VERSIONS' or name='MAX_VERSIONS_PER_RECORD' or name='TABLE_ID_OF_MAX_NUM_VERSIONS');

Note that for scale-out configuration you have to check the master host. If the number of active versions (NUM_VERSIONS) is greater than 8,000,000, it is considered a problem and an overall slowdown of the system can be experienced. Similarly, if the maximum number of versions per record (MAX_VERSIONS_PER_RECORD) exceeds 8,000,000, this should be treated as a problem and a slowdown of accesses to a specific table is expected. Use TABLE_ID_OF_MAX_NUM_VERSIONS and join it against the SYS.TABLES system view to determine the table which is having the problem.

Setting a Memory Limit for SQLStatements

The statement memory limit allows you to set a limit both per statement and per SAP HANA host. To apply these settings you must have the system privilege INIFILE ADMIN. For more details check related SUD HOW Document OPR.HDB.12 - HANA DB User Security Management.

You can protect an SAP HANA system from uncontrolled queries consuming excessive memory by limiting the amount of memory used by single statement executions per host. By default, there is no limit set on statement memory usage but if a limit is applied statement executions that require more memory will be aborted when they reach the limit. To avoid canceling statements unnecessarily you can also apply a percentage threshold value which considers the current statement allocation as a proportion of the global memory currently available.

View the (peak) memory consumption of a statement in M_EXPENSIVE_STATEMENTS.MEMORY_SIZE. To do so, follow parameters needs to be set in global.ini:
enable_tracking = on
memory_tracking = on. You can also create exceptions to these limits for individual users (for example, to ensure an administrator is not prevented from doing a backup) by setting a different statement memory limit for each individual. For this option, additionally, resource tracking must be enabled in the global.ini file.

Memory limit

In the global.ini file set the parameter statement_memory_limit. Set a statement memory limit in GB (integer values only) with a value between 1 and some fraction of the global allocation limit. When the statement memory limit is reached, a dump file is created with 'compositelimit_oom' in the name. The statement is aborted, but otherwise the system is not affected. By default only one dump file is written every 24 hours. The interval can be configured in the memorymanager section of the global.ini file using the oom_dump_time_delta parameter, which sets the minimum time difference (in seconds) between two dumps of the same kind (and the same process).

Memory limit threshold

In the global.ini file set the parameter statement_memory_limit_threshold as a percentage of the global allocation limit (global_allocation_limit).This parameter provides a means of controlling when the statement_memory_limit is applied. If this parameter is set, when a statement is issued the system will determine if the amount of memory it consumes exceeds the defined percentage value of the overall global_allocation_limit parameter setting.

To set a user-specific statement limit and exclude a user from the global limit use the ALTER USER statement as shown here: ALTER USER <user_name> SET PARAMETER STATEMENT MEMORY LIMIT = <gb>

If both a global and a user statement memory limit are set, the user-specific limit takes precedence, regardless of whether it is higher or lower than the global statement memory limit. If the user-specific statement memory limit is removed the global limit takes effect for the user. The value of the parameter is shown in USER_PARAMETERS (like all other user parameters)

To reset a user-specific limit use the CLEAR option: ALTER USER <user_name> CLEAR PARAMETER STATEMENT MEMORY LIMIT

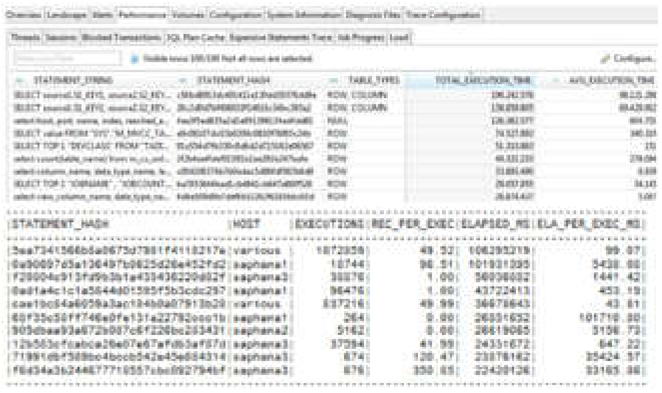
Identification of Critical SQLStatements

A key step in identifying the source of poor performance is to understand how much time is spent in the SAP HANA engine for query execution. By analyzing SQL statements and calculating their response times, you can better understand how the statements affect application and system performance.

Before you are able to analyze and optimize an SQL statement you have to identify the critical SQL statements. We can distinguish between the following scenarios:

SQL statements that have caused problems in the past, SQL statements that are currently causing problems

To determine SQL statements with a particularly high runtime you can check for the top SQL statements in terms of TOTAL_EXECUTION_TIME in the SQL plan cache in SAP HANA Studio.

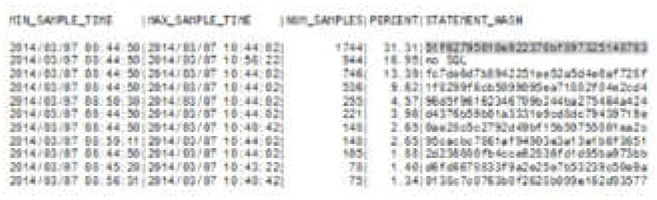


```
|STATEMENT_HASH                  |HOST     |EXECUTIONS|REC_PER_EXEC|ELAPSED_MS|ELA_PER_EXEC_MS|
|3ea7341366b6a0673d7881f4118217e |various  |   1872858|       49.32| 108283219|          99.07|
|6a90097d5a136497b0825d26e452fd2 |saphana1 |     18744|       96.51| 101931395|        5438.00|
|f28004c913fd9b3b1e433436220d82f |saphana3 |     38876|        1.00|  50038032|        1441.42|
|0ad1a4c1c1a5644d01585f5b3cdc297 |saphana1 |     96476|        1.00|  43722413|         453.19|
|cae1bc54a6059a3ac134b0a07913b28 |various  |    837216|       49.99|  36678643|          43.81|
|60f35c50ff746e0fe131a22782ccc1b |saphana1 |       264|        0.00|  26831632|      101718.80|
|905dbae93a672b007c6f226bc283431 |saphana2 |      5162|        0.00|  26619065|        5156.73|
|12b563cfcabca28e07e67efdb3af87d |saphana3 |     37594|       41.99|  24331672|         647.22|
|71891d6f389bc4bccb542e45e684314 |saphana3 |       674|      120.47|  23878162|       35424.57|
|f6434a3b2446777f1655?cbc892794bf|saphana3 |       676|      330.65|  22420126|       33165.86|
```

To determine the top SQL statements that were executed during a dedicated time frame in the past, you can check the SQL plan cache history (HOST_SQL_PLAN_CACHE). You can use the SQL statement: "HANA_SQL_SQLCache_History" (SAP Note 1969700) in order to check for top SQL statements during a specific time frame: You have to specify a proper BEGIN_TIME / END_TIME interval and typically use ORDER_BY = 'ELAPSED', so that the SQL statements with the highest elapsed time from SAP HANA are returned.

The thread sample history (tables M_SERVICE_THREAD_SAMPLES, HOST_SERVICE_THREAD_SAMPLES), if available, can also be used to determine the top SQL statements. You can use the SQL statement:

"HANA_Threads_ThreadSamples_FilterAndAggregation" available from SAP Note 1969700. You have to specify a proper BEGIN_TIME / END_TIME interval and use AGGREGATE_BY = 'STATEMENT_HASH' to identify the top SQL statements during the time frame.



```
MIN_SAMPLE_TIME     |MAX_SAMPLE_TIME     |NUM_SAMPLES|PERCENT|STATEMENT_HASH

2014/03/07 08:44:50|2014/03/07 10:44:02|       1744|  31.31|51f62795010e922370bf897325148783
2014/03/07 08:44:50|2014/03/07 10:56:22|        944|  16.95|no SQL
2014/03/07 08:44:50|2014/03/07 10:44:02|        746|  13.39|fc7de6d7b8942251ee52a5d4e0af728f
2014/03/07 08:44:50|2014/03/07 10:44:02|        536|   9.62|1f8299f6cb5099095ea71882f84e2cd4
2014/03/07 08:50:38|2014/03/07 10:44:02|        253|   4.37|96a9f961623467b9b244ba275484a424
2014/03/07 08:44:50|2014/03/07 10:44:02|        221|   3.96|d4376b59b01a1331e9d6dc7943971be
2014/03/07 08:44:50|2014/03/07 10:40:42|        148|   2.65|0ee28c0c2792d49bf156307350f1aa2s
2014/03/07 08:59:11|2014/03/07 10:44:02|        148|   2.65|90caebc7061ef94903a3e13efb6f3651
2014/03/07 08:44:50|2014/03/07 10:44:02|        105|   1.88|2e1384860fb4cca62836fdfd55ba97366
2014/03/07 08:45:29|2014/03/07 10:43:22|         78|   1.40|a6fd6679833f9a2e25e7b53239c59e8a
2014/03/07 08:56:31|2014/03/07 10:40:42|         75|   1.34|9138c7c0763b0f2625b099e162d93577
```

In this case the SQL statement with hash 51f62795010e922370bf897325148783 is executed most often and so the analysis should be started with it. Often you need to have a look at some more SQL statements, for example the statements related to the next statement hashes fc7de6d7b8942251ee52a5d4e0af728f and 1f8299f6cb5099095ea71882f84e2cd4

In cases where the M_SERVICE_THREAD_SAMPLES / HOST_SERVICE_THREAD_SAMPLES information is not usable you can use the thrloop.sh script to regularly collect thread samples as described in SAP Note 1989031

In case of an out-of-memory (OOM) situation you can determine potentially responsible SQL statements by analyzing the OOM dump file(s) as described in SAP Note1984422

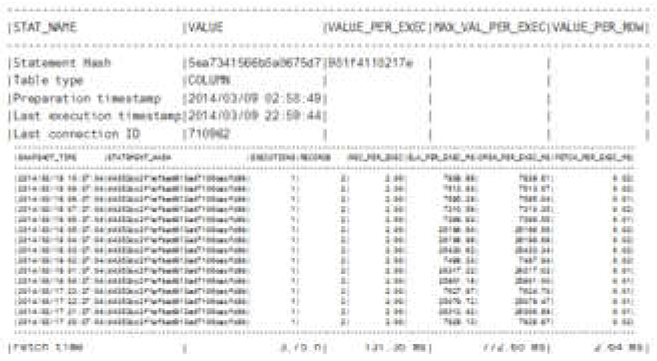
SAP HANA Alert 39 ("Long running statements") reports long running SQL statements and records them in the table _SYS_STATISTICS.HOST_LONG_RUNNING_STATEMENTS. Check the contents of this table to determine details of the SQL statements that caused the alert.

4. Case Study

SQL Plan Cache Analysis

The SAP HANA SQL Plan Cache can be evaluated in detail for a particular statement hash.

The SQL statement: "HANA_SQL_StatementHash_KeyFigures" available in SAP Note 1969700 can be used to check for the SQL Plan Cache details of a specific SQL statement (the related STATEMENT_HASH has to be maintained as input parameter).



```
|STAT_NAME                |VALUE              |VALUE_PER_EXEC|MAX_VAL_PER_EXEC|VALUE_PER_ROW|

|Statement Hash           |5ea7341566b6a0675d7|b51f4118217e  |                |             |
|Table type               |COLUMN             |              |                |             |
|Preparation timestamp    |2014/03/09 02:58:49|              |                |             |
|Last execution timestamp |2014/03/09 22:59:44|              |                |             |
|Last connection ID       |710962             |              |                |             |
[illegible]
|Fetch time               |                   |        3,70 h|       131.30 ms|    772.60 ms|    2.04 ms|
```

The historic execution details for a particular SQL statement can be determined with the SQL statement: "HANA_SQL_StatementHash_SQLCache_History" included with SAP Note 1969700. Also here the appropriate STATEMENT_HASH has to be specified as input parameter.

5. Results

Based on the results of this evaluation you can distinguish the following situations:

If the value for Executions is unexpectedly high, further analysis should be done on the application side in order to check if it is possible to reduce the number of executions.

If the value for Records is unexpectedly high, further analysis should be done on the application side in order to check if it is possible to reduce the number of selected records.

If the value for Cursor duration is very high and at the same time significantly higher than the value for Execution time, you have to check which processing steps are executed on the application side between the individual fetches. A high value for Cursor duration can negatively impact the database in general because open changes may impact the MVCC mechanism.

If the value for Preparation time is responsible for a significant part of the Execution time value you have to focus on optimizing the parsing (for example, sufficient SQL plan cache size, reuse of already parsed SQL statements).

If Execution time is much higher than expected (that can be based on the statement complexity and the number of processed rows), the SQL statement has to be checked more in detail on technical layer to understand the reasons for the high runtime.

6. Detailed Statement Analysis

When you have identified a critical SQL statement and identified its overall key figures from the SQL plan cache analysis you can have a closer look at the actual runtime behavior of the SQL statement. The following tools will be used by the SAP team:

Plan Explanation
Creation of an execution plan Plan Visualizer

Detailed graphical execution plan Temporal breakdown (timeline) available QO Trace
Query optimizer trace
Advanced tool that can be useful to understand the decisions of the query optimizer Particularly helpful to understand column searches
JE Trace
Join evaluation trace
Advanced tool to analyze table join operations Performance trace
Low level recording of key performance indicators for individual SQL statement processing steps Advanced tool that should only be used in collaboration with SAP support
Kernel profiler
Sample based profiling of SAP HANA process activities
Advanced tool that should only be used in collaboration with SAP support

Optimization of Critical SQLStatements
You can improve the general performance of the SAP HANA database by implementing various best practices, design principles, available features, and add-ons. To enhance the performance of the SAP HANA database, the SAP team will do the following: Optimize outlier queries

Queries that sometimes take much longer than expected can be caused by query-external factors (for example, resource bottlenecks) that have to be determined and eliminated.

Check data manipulation commands (DML)

DML operations like INSERT, UPDATE and DELETE can be impacted by lock waits. Improve SQL query design

The performance of your SQL queries can be improved significantly by knowing how the SAP HANA database and SAP HANA engines process queries and adapting the queries accordingly.

Create indexes for any non-primary key columns that are often queried.

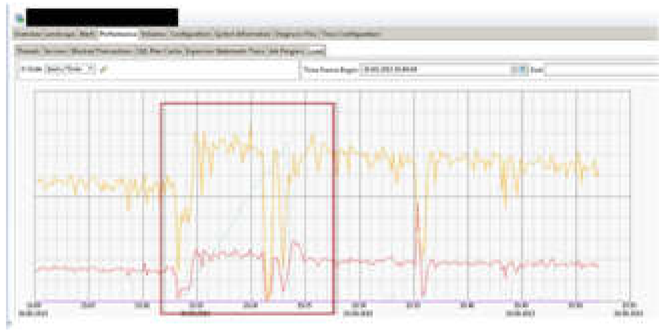
SAP HANA automatically creates indexes for primary key columns; however, if you need indexes for non- primary key columns, you must create them manually. Create virtual star schemas on SAP HANA data by joining attribute views to tables that contain measures within the definition of an analytic view

By using analytic views, SAP HANA can automatically recognize star queries and enable the performance benefits of using star schemas, such as reducing dependencies for query processing. Develop procedures to embed data-intensive application logic into the database. With procedures, no large data transfers to the application are required and you can use performance- enhancing features such as parallel execution. Procedures are used when other modeling objects, such as analytic or attribute views, are not sufficient.

If you use SQL script to create procedures, follow the best practices for using SQL Script. For statistical computing, create procedures using the open source language R.

Download and install the available application function libraries, such as Predictive Analysis Library (PAL) and Business Function Library (BFL). Application functions are like database procedures written in C++ and called from outside to perform data intensive and complex operations.

Scale SAP HANA to improveperformance.
SAP HANA's performance is derived from its efficient, parallelized approach. The more computation cores your SAP HANA server has the better overall system performance is.


[redacted]

You can see that during the period in the red rectangle both CPU consumption and SQL throughput decreased. During that time frame you would look for something that consumed a lot of resources or blocked the statements (locking); just after 15:35 you see that the CPU consumption increases while the SQL throughput decreases. Here, a possible case would be a change in usage: instead of many small, fast SQL statements the workload changed to a few "heavy" (complicated calculation requiring many CPU cycles) SQL statements.

If there was a high statement load in the same period when you experienced the slow execution the root cause is likely a lack of resources. To resolve the situation consider restricting the number of users on SAP HANA, upgrading the hardware, or get in touch with SAP Support if scalability can be improved in this case.

If you did not experience a high statement load during the time frame of the problem, check for background activities:

Delta Merges: Use the monitoring view M_DELTA_MERGE_STATISTICS to check if delta merges happened. In that case try to improve the delta merge strategy to prevent merges happening in phases where users are disturbed

Column Unloads: See the Monitoring View M_CS_UNLOADS to look for signs of column unloads. If a column used in the problematic statement had to be loaded before execution, the execution itself will take significantly longer.

Savepoints: Savepoints consume resources and write-lock the database during their critical phase. Check M_SAVEPOINTS and look for save points during the time frame of the problem. If a save point slowed down your execution, the chance of having the same problem again is very low. If this is happening very often the service team will engage accordingly.

Performance of SQL Series can be improved significantly by knowing how the SAP HANA database and SAP HANA engines process queries and adapting your queries accordingly.

As a general guideliñe for improving SQL query performance, SAP recommends to avoid operations that are not natively supported by the various SAP HANA engines. Such operations can significantly increase the time required to process queries. In addition, the following specific recommendations may help improve the performance of your SQL queries:

Avoid calculations in queries.

If two columns are frequently compared by queries, ensure the two columns have the same data type.

For columns of different types, SAP HANA uses implicit type casting to enable comparison. However, implicit type casting has a negative effect on performance. If you cannot ensure the two columns have the same type from the beginning, one of the following steps can improve performance:

If possible, change the type of value as this has less cost than changing the type of column. Consider adding a generated column. A generated column improves query performance at the expense of increased insertion and update costs
Avoid join predicates that do not have the equal condition. Join predicates connected by OR, Cartesian product, and join without join predicates are not natively supported

Avoid using filter predicates inside outer join predicates because they are not natively supported. You can rewrite such filter predicates into equijoin predicates using a generated column.

Avoid cyclic joins because cyclic outer joins are not natively supported and the performance of cyclic inner joins is inferior to acyclic inner joins.

Avoid using OR to connect EXISTS or NOT EXISTS predicates with other predicates.

If possible, use the NOT EXISTS predicate instead of NOT IN. The NOT IN predicate is more expensive.

Avoid using the UNION ALL, UNION, INTERSECT and EXCEPT predicates because they are not natively supported.

For multiple columns involved in a join, create the required indexes in advance. SAP HANA automatically creates indexes to process joins; however, if the queries are issued by an update transaction or a lock conflict occurs, the indexes cannot be created, which may cause performance issues.

Create indexes on non-primary key columns to enhance the performance of some queries using the index adviser. SAP HANA automatically creates indexes for all primary key columns. Indexing the primary key columns is usually sufficient because queries typically put filter conditions on primary key columns. When filter conditions are on non-key fields and tables have many records, creating an index on the non-primary key columns may improve the performance.

Example:
The SAP Service team can Check whether there is an index for a column, in the Plan Visualizer, in the properties of a column, see the Inverted Index entry. Alternatively, you can also see the system view M_INDEXES. It is possible to create indexes on non-primary key columns to enhance the performance of some queries, particularly highly selective queries on non-primary key columns. This can be checked in the service as well.

7. Limitations and Future Study

Memory is a vast topics and issues related to them can't be predicted. It depends behavior of Hardware, Database, OS and applications. Few cases are covered in this article. In another article few more cases will be analyzed. Hard vendors does provide SAS for SAP applications but limited to its sizes. Also, disk, CPU, I/O case studies, delta, main merge, MVCC will be covered.

8. Conclusion

SAP HANA is a in memory complex database which requires to analyze thoroughly for large enterprise database systems. Being columnar Database for OLTP and OLAP, processing speed is much higher than traditional RDMS databases. A BI report used to take 10 minutes to process on a Oracle database, consuming less resources and less than 1 minute (50 milli seconds)

Declarations

Ethics approval and consent to participate: Not Applicable

Consent for publication: All authors have consent to submit this paper to Journal of Cloud Computing. Also, we confirm that this paper or any part of this paper did not submit any where

Availability of data and materials: Not Applicable

Competing interests: Not Applicable

Funding: Not Applicable

References

[1] What is SAP HANA. [Online]. Available at: https://www.ibm.com/topics/sap-hana
[2] SAP HANA Installing and administering. SAP TRAINING. [Online]. Available at: https://training.sap.com/course/ha200-sap-hana-installing-and-administering-classroom-019-g-en/?
[3] SAP HANA Administration Guide. SAP Help. [Online]. Available at: https://help.sap.com/docs/SAP_HANA_PLATFORM/6b94445c94ae495c83a19646e7c3fd56/bde79b28bb5710

149d6eee5e 75fe7f17.html

[4] Memory Usage in the SAP HANA Database. SAP Help. [Online]. Available at: https://help.sap.com/docs/SAP_HANA_PLATFORM/6b94445c94ae495c83a19646e7c3fd56/bde79b28bb5710149d6eee5e 75fe7f17.html

[5] How to Start and Stop an Instance of SAP HANA, SAP PRESS. [Online]. Available at: https://blog.sap-press.com/how-to-start-and-stop-an-instance-of-sap-hana#:~:text=With%20SAP%20HANA%20Studio%201%20Starting%20Choose%20Start,to%20look%20at%20the%20instance%20trace%20files.%20

[6] Sap-hana-tmpfs.service, Github. [Online]. Available at: https://github.com/aws-samples/aws-sap-hana-fast-restart-scripts/blob/main/sap-hana-tmpfs.service

[7] Sap-hana-tmpfs.sh, Github. [Online]. Available at: https://github.com/aws-samples/aws-sap-hana-fast-restart-scripts/blob/main/sap-hana-tmpfs.sh

[8] SAP HANA on AWS. Amazon Doc. [Online]. Available at: https://aws.amazon.com/sap/solutions/saphana/

[9] Help Portal Documentation, SAP HANA Platform. SAP Help. [Online]. Available at: https://help.sap.com/docs/SAP_HANA_PLATFORM

[10] Gilbert Wong, Data Management for Technical Tables for SAP S/4HANA, SAP Blogs. [Online]. Available at: https://blogs.sap.com/2021/02/09/data-management-for-technical-tables-for-sap-s-4hana/

[11] Big Data, Google Trends. [Online]. Available at: https://trends.google.com/trends/explore?date=all&q=Big%20Data

[12] Richard Bremer (Author), Lars Breddemann (Author), SAP HANA Administration, SAP PRESS [Online]. Available at: https://www.sap-press.com/sap-hana-administration_3506/

[13] TADM10_1 & TADM10_2 SAP Books, Technical Implementation and Operation I of SAP S/4HANA and SAP Business Suite

[14] TADM_51, SAP NetWeaver AS - DB Operation (Oracle), [Online}. Available at: https://cdn20.training.sap.com/cdn/course-pdf/TADM51_EN_Col15_ILT_FV_CO_A4.pdf/G/EN/TADM51/015

[15] Memory Usage in the SAP HANA Database. SAP Help. [online]. Available at: https://help.sap.com/docs/SAP_HANA_PLATFORM/6b94445c94ae495c83a19646e7c3fd56/bde79b28bb5710149d6eee5e 75fe7f17.html?version=2.0.04

[16] What is SAP HANA. Sap.com. [online]. Available at: https://www.sap.com/products/technology-platform/hana/what-is-sap-hana.html