International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V72I11P105

Volume 72 Issue 11, 31-38, November 2024

Original Article

SAP HANA Workload Management: A Comprehensive Study on Workload Classes

Puneet Aggarwal[1], Amit Aggarwal[2]

[1]*Department of Information & Technology, Deloitte LLP, Texas, United States of America.*
[2]*Department of Information & Technology, TCS, Michigan, United States of America.*

[1]*Corresponding Author : erpuneetaggarwal@gmail.com*

Received: 20 September 2024 Revised: 23 October 2024 Accepted: 12 November 2024 Published: 29 November 2024

Abstract *- This study investigates workload management challenges in SAP HANA's in-memory columnar storage, specifically focusing on the complexities of handling mixed OLTP and OLAP workloads. Traditional database management approaches often fail to address SAP HANA's unique architecture, which combines high-speed transaction processing with real-time analytics. Drawing on research into mixed workload optimization (May et al., 2015) and SAP HANA's architectural evolution (Färber et al., 2017), this paper explores workload throttling, prioritization, and real-time monitoring strategies to optimize resource allocation. Our findings offer practical guidance for database administrators managing critical business applications with high data demands, such as finance and supply chain. Additionally, we examine tools and techniques for performance tracing in third-party applications like Tableau (Tableau Software, 2023), providing an integrated perspective on maintaining optimal SAP HANA performance under peak load conditions.*

Keywords *- SAP HANA Workload Management, In-Memory Database, Mixed Workloads, OLTP and OLAP, Resource Allocation, Workload Throttling, Real-Time Monitoring, Database Performance Optimization, Admission Control, Workload Classes, Tableau Integration.*

1. Introduction

In today's data-intensive business landscape, organizations rely on robust and efficient database systems to process and analyze large volumes of information in real time. With its in-memory, columnar data storage, SAP HANA is a preferred solution for enterprises needing high-performance data handling. However, as these data demands grow, managing workloads on SAP HANA poses unique challenges. High-demand tasks—such as processing complex transactions, generating comprehensive reports, and enabling real-time analytics—can significantly strain CPU and memory resources, potentially causing performance bottlenecks. Traditional workload management techniques are often insufficient for SAP HANA's architecture, which is designed to balance concurrent OLAP (Online Analytical Processing) and OLTP (Online Transaction Processing) workloads within a single system. To address these challenges, this study aims to develop and assess workload management strategies tailored to SAP HANA's specific needs, focusing on the following objectives:

- Analyze SAP HANA Workload Patterns: Understand how mixed OLTP and OLAP workloads impact CPU and memory usage, identifying key areas of resource contention.
- Implement Workload Throttling: Explore workload throttling techniques that regulate resource use, preventing system overloads during peak processing times.
- Prioritize Workloads and Allocate Resources: Establish methods to prioritise critical tasks and receive adequate resources, minimizing the risk of system slowdowns.
- Enhance Real-Time Monitoring and Admission Control: Assess the use of performance monitoring tools and introduce proactive workload management strategies to optimize resource allocation and maintain system stability.

This study provides database administrators with practical strategies to enhance SAP HANA's performance and reliability in high-demand environments by achieving these objectives. The research is especially relevant for sectors where data processing demands are intensive, such as finance, retail, and supply chain management, offering insights that can help sustain peak performance even during periods of heavy use.

1.1. Importance of the Study

As organizations navigate an increasingly data-driven environment, processing large volumes of data in real time has

become critical. For industries like finance, retail, manufacturing, and supply chain management, efficient data handling is essential for maintaining competitive advantage and operational stability. SAP HANA, known for its in-memory architecture, is well-suited to meet these demands; however, its performance can be hindered when managing complex, high-demand workloads. Addressing these challenges through effective workload management is essential to harness the full potential of SAP HANA's capabilities.

Specific examples of data growth challenges include:

- Finance: Processing millions of transactions daily, generating detailed financial reports, and supporting real-time fraud detection.
- Retail: Managing large-scale inventory, processing high volumes of sales transactions, and analyzing customer behavior for personalized marketing.
- Manufacturing: Planning production schedules, ensuring quality control, and coordinating logistics.
- Supply Chain: Coordinating demand and supply planning, tracking shipments, and optimizing inventory across multiple locations.

When computational resources become strained, prioritizing and managing workloads effectively ensures critical business tasks maintain priority while system performance remains stable. This study contributes to the field by developing and evaluating workload management strategies designed specifically for SAP HANA's in-memory processing model, allowing database administrators to improve performance under heavy load conditions. By doing so, it provides insights and practical recommendations that support both operational efficiency and business continuity.

2. Workload Management

In SAP HANA, workload management is essential to ensure that system resources are allocated efficiently to handle varying demands from different tasks. Administrators can avoid resource contention, reduce bottlenecks, and maintain system stability by actively managing workloads. Workload management strategies in SAP HANA focus on optimizing CPU and memory usage, prioritizing tasks, and ensuring that both real-time transactions and analytical queries perform effectively. Understanding the types of workloads in SAP HANA is the first step toward implementing tailored management strategies for optimal system performance.

2.1. Types of Workload in SAP HANA

SAP HANA handles diverse workload types, each with distinct requirements and impacts on system resources. Understanding these workloads is essential for optimizing performance and ensuring that the database can handle concurrent tasks without performance degradation. The primary workload types in SAP HANA include:

2.1.1. OLAP Workload

Used primarily for reporting and analytics, OLAP (Online Analytical Processing) workloads are common in data warehousing and business intelligence applications, such as reporting in BW (Business Warehouse) systems or SAP Analytics Cloud. These tasks are typically resource-intensive, requiring significant memory allocations and parallel execution capabilities to process large datasets effectively.

2.1.2. OLTP Workload

OLTP (Online Transaction Processing) workloads focus on real-time transaction processing, handling high volumes of smaller, rapid database operations. Commonly used in ERP systems like SAP S/4HANA and ECC 6.0, OLTP workloads require low-latency access to data and consistent CPU availability to support real-time processing with minimal delay.

2.1.3. Mixed Workload

Many modern systems operate with a mixed workload environment, combining both OLAP and OLTP processes. For instance, ERP systems often integrate transactional processing with analytical reporting functions. Mixed workloads place high demands on both memory and CPU resources, requiring sophisticated workload management to balance competing tasks and maintain overall system performance.

2.1.4. Internal Workloads

Internal workloads are generated by SAP HANA's system processes, including tasks like merges, backups, and savepoints. Though not directly related to user-initiated tasks, these processes are essential for database maintenance and integrity. Internal workloads consume CPU and memory resources, usually during scheduled maintenance windows, to ensure consistent system performance and reliability. By categorizing these workload types, administrators can better allocate resources and apply targeted workload management strategies to balance system demands effectively.

2.2. Ways to control HANA Workloads

Effective workload management in SAP HANA requires the optimal allocation and utilization of critical system resources, primarily CPU threads and memory. Administrators can manage resource consumption, improve performance, and maintain system stability by implementing strategic controls. Key approaches include:

2.2.1. Memory Analysis and Global Limit

Efficient memory management is crucial for maintaining SAP HANA's performance, especially under heavy workload conditions. SAP HANA provides the global_allocation_limit parameter, which sets a limit on the total memory the system can utilize, ensuring that memory-intensive tasks do not overwhelm available resources. By default, this limit is calculated based on the system's physical memory—typically

set to 90% of the first 64GB and 97% for each additional GB of memory. However, administrators can adjust these values according to specific workload needs. In addition to the global allocation limit, SAP HANA provides detailed memory analysis tools to monitor how memory is being used across different workloads. Administrators can identify potential bottlenecks and adjust allocations to avoid performance issues by assessing real-time memory usage. This proactive memory management strategy enables administrators to balance memory demands for both OLTP and OLAP tasks effectively, reducing the risk of memory saturation and ensuring smooth system operations.

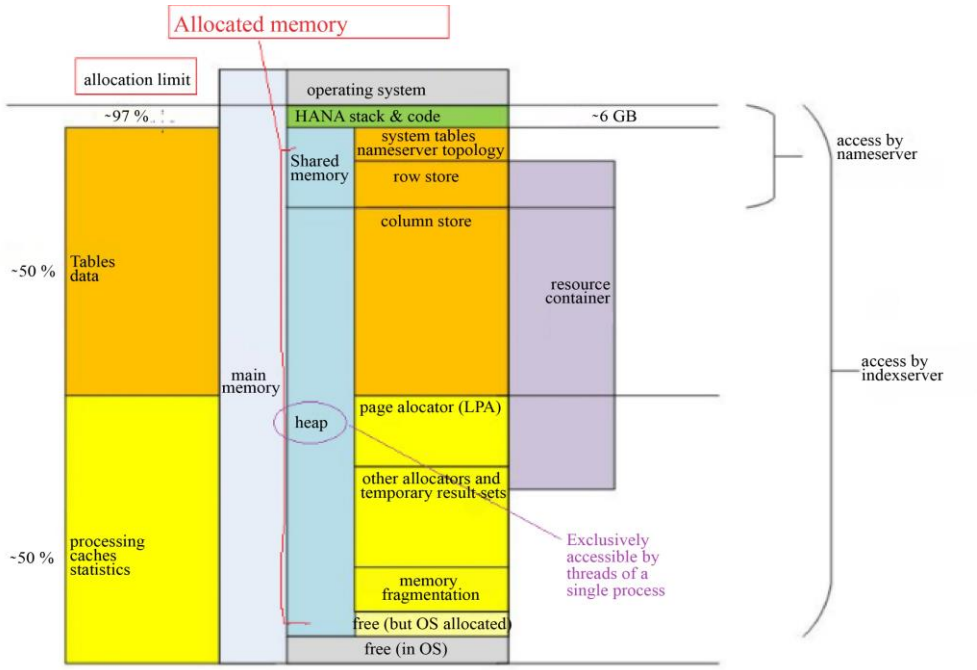

Fig. 1 HANA memory areas (Allocation and heap memory limits)

2.2.2. CPU Threads

SAP HANA relies on multiple CPU threads to execute tasks, leveraging a multi-threaded architecture to handle diverse and concurrent workloads. Each SAP HANA process, such as the indexserver and xsengine, operates with its own set of threads to manage various operations efficiently. SAP HANA threads are divided into two main types:

SQLWorker Threads

Primarily responsible for handling OLTP-like (Online Transaction Processing) statements, SQLWorker threads are typically used for straightforward, low-latency operations executed in a single-threaded mode. When SQL statements are more complex or become blocked, they are delegated to JobWorker threads for execution. In the monitoring view M_SERVICE_THREADS, these threads appear as SqlExecutor.

JobWorker Threads

These threads handle complex or parallelized OLAP (Online Analytical Processing) operations. By default, the number of JobWorker threads corresponds to the number of CPU cores available to the process. However, this count may be adjusted if many SQLWorker threads consume CPU resources concurrently. Administrators can configure JobWorker behavior through parameters to align with workload demands. In M_SERVICE_THREADS, JobWorker threads are listed as JobWorker. To further manage thread allocation, administrators can adjust several thread-related parameters. For instance, the max_concurrency parameter controls the number of concurrent statements allowed, while max_sql_executors specifies the limit on threads that can process SQL requests simultaneously.

By carefully configuring these parameters, administrators can prevent excessive CPU contention, balance processing loads across threads, and improve overall system performance. Basic thread calculation is explained using a screenshot from a laptop to make it easier for everyone; on linux we can check using lscpu or other commands.

No of Threads = No of Socket * No Cores * Logical Processor = 768 Threads

- Socket: A physical connection point on a motherboard where a CPU can be inserted.
- Core: A processing unit within a CPU that can execute instructions independently.
- Logical Processor: A software-based simulation of a physical core, often enabled by hyper-threading technology.

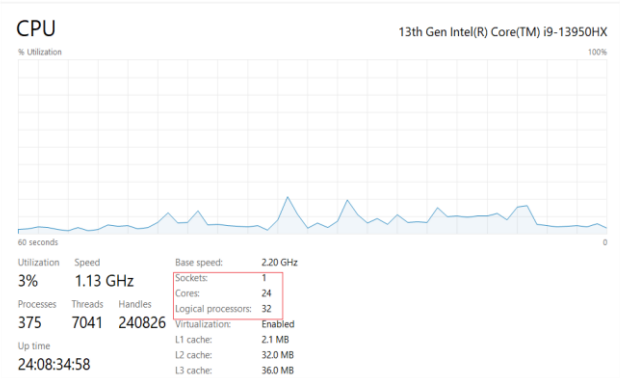

Fig. 2 CPU thread count (Sockets * Cores * processor)

Peak Load Admission Control

To manage high system utilization and prevent overloads, SAP HANA includes an admission control feature that regulates database request execution during peak load times. This feature, introduced in SAP HANA 2.0, enables administrators to configure thresholds for queuing and rejecting requests based on current CPU and memory utilization levels. When system resources reach these set limits, admission control can either queue incoming requests for later execution or reject them outright, ensuring that critical tasks already in progress receive the necessary resources to complete successfully. Key parameters include queue_cpu_threshold and queue_memory_threshold, which define the levels at which requests are queued, as well as reject_cpu_threshold and reject_memory_threshold, which specify when requests are rejected to prevent system overload. For instance, if CPU utilization reaches a defined threshold, requests are temporarily held in a queue until resource availability improves. Rejected requests fail with error codes like "616: rejected by workload class configuration" or "1038: service temporarily unavailable," providing administrators with clear indicators of system status. By configuring admission control settings, administrators can maintain system stability during times of high demand, prevent resource contention, and ensure that essential workloads maintain priority. This approach provides a buffer against sudden spikes in demand, allowing SAP HANA to continue operating efficiently under even the most resource-intensive conditions.

2.3. Workload Management Using Workload Classes

In SAP HANA, workload management is achieved through the use of workload classes, which allow administrators to categorize and manage resources for specific types of tasks. Each workload class can have defined limits and priorities, enabling the system to dynamically adjust resource consumption based on workload requirements. SAP HANA administrators can effectively control resource allocation and ensure optimal system performance by mapping tasks to specific workload classes.

Key properties of workload classes include:

1) Priority: This parameter assigns priority levels to statements within a workload class, helping the system prioritize critical tasks during execution. Priority levels range from 0 (lowest priority) to 9 (highest), with a default of 5.
2) Statement Thread Limit: This property limits the number of threads that can be used by a single statement, preventing excessive parallelism. It can range from 0 (no limit) to the number of logical cores in the system.
3) Statement Memory Limit: This property restricts the amount of memory that a single statement within the workload class can use, helping to prevent memory overuse by any single task. It can be set to values up to the global memory allocation limit.
4) Total Statement Thread Limit: This property specifies a cumulative thread limit for all statements executing within the workload class, balancing thread usage across multiple tasks. This is particularly useful for controlling overall system thread usage when multiple active statements exist.
5) Total Statement Memory Limit: This property sets a cumulative memory limit for all statements within the workload class, helping prevent memory saturation. It ensures that tasks within a workload class do not collectively consume more than a set amount of memory.
6) Statement Timeout: This parameter defines the maximum execution time for each query. If the query fails to complete within this time frame, it will be terminated, triggering an error (ERR_API_TIMEOUT).
7) Write Transaction Lifetime: This property limits the maximum time an uncommitted write transaction can remain open, ensuring that long-running transactions do not lock resources indefinitely.
8) Idle Cursor Lifetime: This parameter limits the time a cursor can remain idle. If a cursor is inactive beyond the specified duration, it is terminated to free up system resources.

Thread Limit Parameters: These parameters allow administrators to control thread allocation per workload class, helping balance CPU usage and prevent contention:

- max_concurrency: Sets the maximum number of concurrent statements allowed on the system.
- max_concurrency_dyn_min_pct: Defines the minimum percentage of maximum concurrency that must be dynamically maintained.
- default_statement_concurrency_limit: Specifies the default limit for concurrent statements in a workload class.
- default_statement_concurrency_limit_rel: Sets the default relative limit based on the system's overall concurrency limit.
- default_statement_concurrency_max_limit: Establishes the maximum limit for concurrent statements in a workload class.

- max_concurrency_hint: Provides a hint for the maximum concurrency level to which a specific statement should adhere.
- max_concurrency_hint_dyn_min_pct: Indicates the minimum percentage of the max concurrency hint to be dynamically maintained.
- max_sql_executors: Specifies the maximum number of SQL executor threads for normal SQL request processing.

By configuring workload classes and their properties effectively, SAP HANA administrators can ensure that essential tasks receive appropriate resources, maintaining optimal performance across different workloads.

2.4. Admission Control in Workload Classes

Admission control in SAP HANA workload classes enables administrators to manage system performance by setting resource utilisation thresholds, ensuring critical operations can continue even under high-demand conditions. This feature prevents the system from overloading by controlling SQL request admission based on CPU and memory usage.

Key admission control parameters include:

1) Admission Control Reject CPU Threshold and Admission Control Reject Memory Threshold: These parameters define the threshold levels (0-100%) at which SQL requests are rejected. When CPU or memory usage reaches the specified threshold, new SQL requests are immediately rejected with an error message (e.g., "616: rejected by workload class configuration").

2) Admission Control Queue CPU Threshold and Admission Control Queue Memory Threshold: These parameters set the levels (1-100%) at which SQL requests are queued instead of being rejected outright.

Queued requests will be executed once resource usage drops below the threshold. If a queued request cannot be executed within the specified queue timeout, it is rejected with an error message (e.g., "1038: service temporarily unavailable").

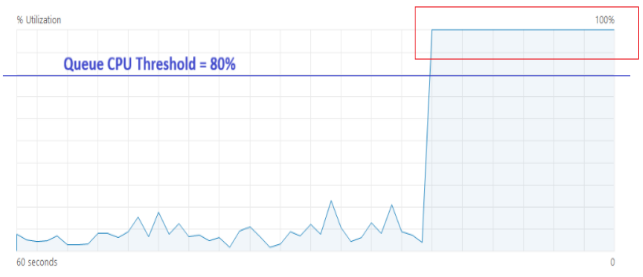

Fig. 3 CPU threshold limit

For example, an administrator might configure the CPU queue threshold at 80% and the CPU reject threshold at 90%. Under this setup, when CPU utilization reaches 80%, incoming SQL requests are placed in a queue. If CPU usage rises to 90%, new requests are rejected outright to maintain system stability.

For other admission control parameters, check 2186744. Historic admission control details can be determined via SQL: "HANA_LoadHistory_Services

Example Configuration for Admission Control:

```
CREATE WORKLOAD CLASS "CriticalWorkloadClass"
SET 'PRIORITY' = '9',
  'STATEMENT MEMORY LIMIT' = '4',
  'STATEMENT THREAD LIMIT' = '10',
  'ADMISSION CONTROL QUEUE CPU THRESHOLD' = '75',
  'ADMISSION CONTROL QUEUE MEMORY THRESHOLD' = '85',
  'ADMISSION CONTROL REJECT CPU THRESHOLD' = '90',
  'ADMISSION CONTROL REJECT MEMORY THRESHOLD' = '95';
```

By configuring admission control, SAP HANA administrators can better manage system loads, ensure that essential processes receive priority, and maintain overall stability during peak usage times. This feature provides a proactive approach to workload management, allowing critical tasks to proceed without disruption while protecting the system from excessive load.

2.5. Query Timeout Precedence

In SAP HANA workload management, different timeout settings can be applied at multiple levels, such as the global setting, workload class level, and individual statements. The system determines the effective timeout based on the smallest valid timeout value across these settings, ensuring that query execution is limited to the most restrictive time threshold set.

The following table illustrates how timeout precedence works:

Settings	Value	Effective Value
Query Timeout (global setting)	30	
statement_timeout (ini file)	20	20
STATEMENT TIMEOUT (Workload class)	15	15*
No matching workload class		0 (Disabled)

In this example, the effective timeout value would be 15 seconds, as it is the smallest timeout value defined within the workload class. Other values are either higher or ignored based on precedence rules. If a query exceeds this threshold,

it will automatically timeout, ensuring that long-running queries do not monopolize resources or negatively impact system performance.

This precedence structure allows administrators to enforce strict time limits for specific workload classes while maintaining more lenient settings at the system level. By configuring timeout parameters strategically, SAP HANA can ensure efficient query execution and balanced resource usage across all workloads.

2.6. Workload Key Attributes

Key Workload Attributes in SAP HANA define the structure and hierarchy of processes within a session. Each workload can be broken down into the following components:

- User Connection: Initiates the workload, creating a unique connection ID.
- Session: A combination of the connection, thread, SQL statement, and transaction. Multiple sessions may stem from a single user connection, each operating independently.
- Thread: Represents the worker process on the SAP HANA side. Threads, such as SQLWorker and JobWorker threads, execute the SQL statements and handle specific tasks within the session.

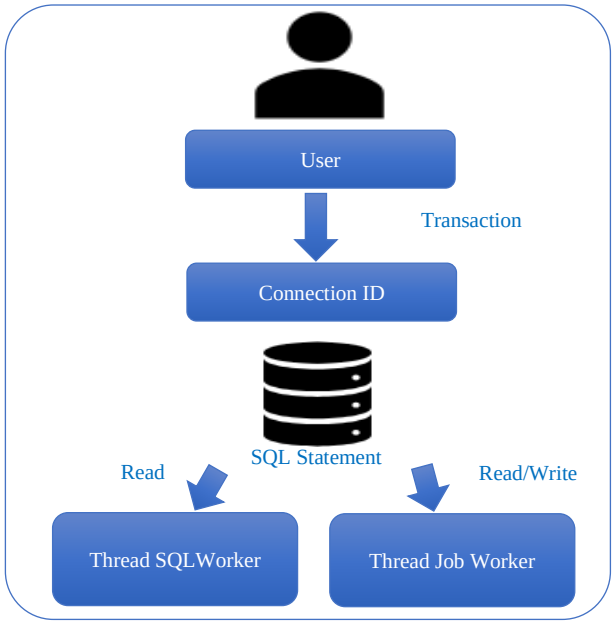

Fig. 4 Flow of user sessions to connection to threads

The above hierarchy says the parameters apply universally across all sessions, thanks to the default class mapping (SYS_DEFAULT). User parameters will take precedence over HDB parameters (*.ini). Workload classes will override both HDB and user parameters. Ultimately, statement hints will have the highest priority, superseding all others.

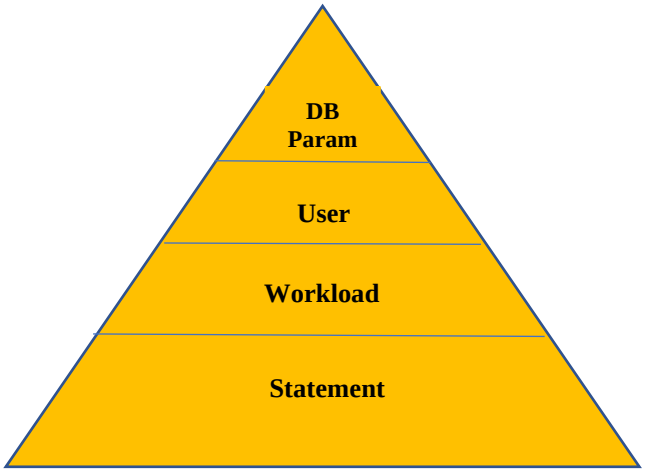

Fig. 5 Hierarchy key parameters

2.7. Analyze and Control SAP Workloads

Analyzing SAP Workloads involves using monitoring views and tools to identify performance bottlenecks and optimize system usage. SAP HANA provides various views to help administrators track activity and evaluate workload patterns:

- M_ACTIVE_STATEMENTS: Displays currently active SQL statements, enabling administrators to monitor real-time workloads and identify heavy or resource-intensive tasks.
- M_PREPARED_STATEMENTS: Lists prepared SQL statements stored within the system, helping to optimize frequently used queries and save on preparation time.
- M_EXPENSIVE_STATEMENTS: Identifies high-cost statements that consume significant resources, allowing administrators to adjust workload classes or apply specific limits to control resource usage.

Additional views, such as M_SESSION_CONTEXT and M_SERVICE_THREADS, offer insights into session-specific variables and active service threads, providing a comprehensive perspective on workload activity. Administrators can also refer to SAP Note 2215929 for setting session variables and workload classes to fine-tune performance.

2.7.1. Controlling HANA Workloads

Controlling HANA Workloads on a User Level allows administrators to apply workload management settings based on specific user profiles. By tailoring workload class settings and resource limits for individual users, SAP HANA ensures that high-priority tasks have adequate resources while balancing system load. Administrators can set parameters like:

- Max_Concurrency: Defines the maximum concurrent tasks a user can execute.
- Statement_Timeout: Limits execution time for user-initiated statements, helping prevent long-running queries from impacting system performance.

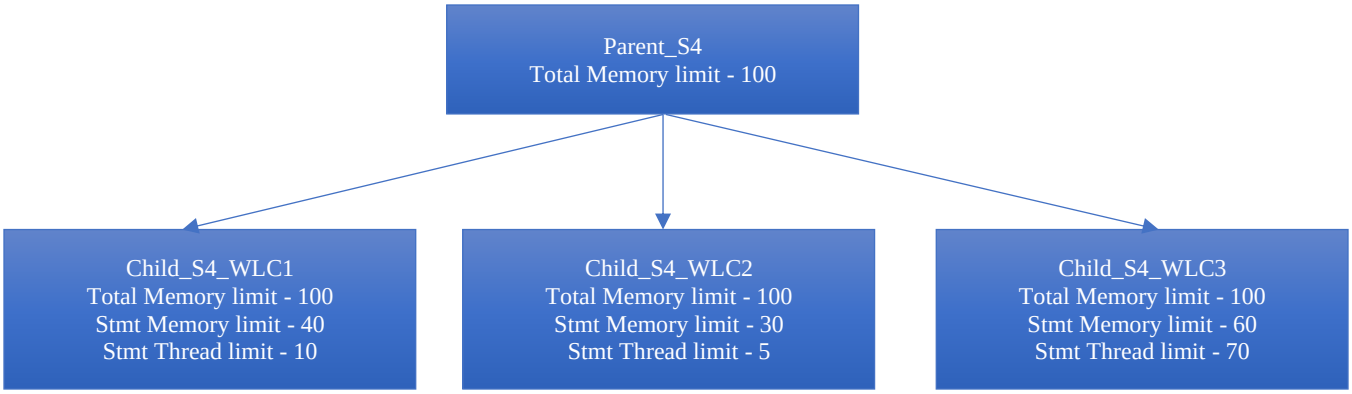

Fig. 6 Hierarchies of workload classes

Grouping workloads based on their characteristics and resource requirements helps manage them more effectively. This involves categorizing tasks into groups such as transactional workloads, analytical workloads, peak workload events, and industry-specific workloads. By grouping workloads, administrators can apply tailored management strategies to each group, ensuring optimal performance and resource utilization. By implementing user-specific controls, SAP HANA offers a more granular approach to workload management, ensuring each user's tasks align with available resources.

2.7.2. Hierarchies of Workload Classes

Hierarchies of Workload Classes enable SAP HANA to organize workload management through a multi-level structure, where different resource limits apply to classes within a hierarchy. Key aspects include:

- Inherited Limits: Child processes inherit cumulative memory limits from parent workload classes, ensuring that individual limits align with overall resource constraints.
- Prioritization: Workload classes are prioritized based on application needs, enabling critical workloads to be processed first without compromising resources for lower-priority tasks.

Administrators can efficiently manage resources across complex environments with multiple processes and user demands by leveraging workload class hierarchies. All child processes will inherit the total memory limit. Each child can have its own individual statement limits (for memory and threads), but the total limit applies collectively. This means that all child processes together share a 100GB limit, rather than each child having an aggregated limit of 100 GB.

2.7.3. Monitoring Views for Workload Classes

SAP HANA provides several Monitoring Views to track workload classes and manage resource utilization:

- WORKLOAD_CLASSES: Displays defined workload classes, along with their memory, thread, and priority configurations.
- WORKLOAD_MAPPINGS: Shows the mappings of workloads to their respective classes, allowing administrators to verify and modify class assignments.
- M_ACTIVE_STATEMENTS and M_EXPENSIVE_STATEMENTS: Track active and resource-intensive statements essential for identifying high-demand processes.
- M_CONNECTIONS and M_SERVICE_THREADS: Offer details on current connections and active threads, giving insights into session-level and thread-level resource usage.

These monitoring views enable administrators to keep track of resource consumption across different workloads, ensuring that system resources are allocated efficiently and aligned with workload priorities.

3. Conclusion

This study on SAP HANA workload management highlights the importance of strategically managing resources to support diverse and intensive workloads in high-demand environments. By implementing workload throttling, workload class mappings, admission control, and dynamic monitoring, SAP HANA administrators can ensure optimal performance, even when processing complex OLTP and OLAP tasks concurrently. By prioritizing critical workloads and proactively managing CPU and memory usage, the techniques outlined provide a comprehensive approach to maintaining SAP HANA's stability and efficiency.

Key findings from this study underscore that:

1. Performance Optimization: Effective workload management significantly enhances SAP HANA's ability to handle high transaction volumes and real-time analytics without resource contention.
2. Granular Control with Workload Classes: Using workload classes, administrators can tailor resource allocation to specific applications or user groups, ensuring that critical processes receive priority.
3. Dynamic Resource Adjustment: Admission control and monitoring views provide insights into resource usage

patterns, allowing for dynamic adjustments in response to changing workload demands.

These strategies enable organizations to fully leverage SAP HANA's capabilities for critical business applications, enhancing productivity and supporting key operations in finance, retail, manufacturing, and supply chain management.

Future Work

While this study provides foundational strategies for managing workloads in SAP HANA, there are areas for further exploration that could enhance workload management practices. Future research could focus on the following:

1. Advanced Machine Learning for Workload Prediction: Integrating machine learning models to predict workload patterns could allow SAP HANA to adjust resources proactively. Predictive analytics could identify peak usage times or potential bottlenecks, allowing for preemptive resource allocation adjustments.
2. Real-Time Dynamic Workload Management: Enhancing real-time workload adjustment capabilities to automatically adapt to fluctuating resource demands across OLTP and OLAP tasks. This would include dynamically fine-tuning admission control and workload classes based on instantaneous load and system conditions.
3. Expanded Workload Classification: Further categorizing workloads based on specific business functions or user groups could enable more precise resource control, aligning workload management closely with business objectives.
4. Security and Compliance Considerations: As data security and regulatory requirements grow, the research could explore workload management methods that integrate security policies, ensuring that sensitive workloads are prioritized and protected under strict access and compliance controls.

Future work in these areas would provide SAP HANA administrators with even more refined tools and strategies for effective workload management, meeting the evolving needs of organizations and further enhancing SAP HANA's role in high-stakes, data-intensive environments.

References

[1] What's New in SAP HANA 2.0 SPS 07 in Administration and Monitoring, SAP Community Blog, 2023. [Online]. Available: https://community.sap.com/t5/technology-blogs-by-sap/what-s-new-in-sap-hana-2-0-sps-07-in-administration-and-monitoring/ba-p/13554753

[2] Managing Workload with Workload Classes, SAP Help Portal, 2023. [Online]. Available: https://help.sap.com/docs/SAP_HANA_PLATFORM/6b94445c94ae495c83a19646e7c3fd56/5066181717df4110931271d1efd84cbc.html

[3] FAQ: SAP HANA Workload Management, SAP Note 2222250, 2023. [Online]. Available: https://userapps.support.sap.com/sap/support/knowledge/en/2222250

[4] FAQ: SAP HANA Memory, SAP Note 1999997, 2023. [Online]. Available: http://sapway.com/WILLSYS/HANA_Note/1999997-FAQ%20SAP%20HANA%20Memory.pdf

[5] FAQ: SAP HANA Threads and Thread Samples, SAP Note 2114710, 2023. [Online]. Available: https://userapps.support.sap.com/sap/support/knowledge/en/2114710

[6] HANA Workload Management Deep Dive – Part III, SAP Community Blog, 2023. [Online]. Available: https://community.sap.com/t5/technology-blogs-by-members/hana-workload-management-deep-dive-part-iii/ba-p/13568799#mapping

[7] Managing Workload with Workload Classes in SAP HANA, SAP HANA Administration Guide, 2023. [Online]. Available: https://help.sap.com/doc/eb75509ab0fd1014a2c6ba9b6d252832/2.0.03/en-US/SAP_HANA_Administration_Guide_en.pdf

[8] Memory Management and CPU Thread Allocation, SAP HANA Administration Guide, 2023. Retrieved from https://help.sap.com/doc/eb75509ab0fd1014a2c6ba9b6d252832/2.0.07/en-US/SAP_HANA_Administration_Guide_en.pdf