1849392 - HANA alerts related to locks, hanging sessions and long runners

Version 1 Validity: 20.04.2013 - active

Symptom

Statisticserver alerts are triggered on HANA side:

HANA alert 39: Long-running statements

HANA alert 42: Long-running cursors

HANA alert 48: Long-running uncommitted write transactions

HANA alert 42: Long-running blocking situations

Environment

HANA

Resolution

These alerts are also reported into the alert trace of the indexserver (indexserver_alert_<host>.trc):

long running uncommitted write transaction detected: CONNECTION ID = 200647, HOST = hanadb:30003, CLIENT HOST = hanaci, CLIENT PIL TRÂNSACTION ID = "902", TRANSACTION TOTAL EXECUTED TIME = 6321 sec

Language English

The long running uncommitted write transaction possibly incurs performance degradation of HANA database. Please commit/rollback a transacti application or kill the connection by "ALTER SYSTEM DISCONNECT SESSION '202982'

As a common quality, these alerts always contain information on the responsible client process. Before any action can be done (commit/rollback/kill), investigation has do be conducted on what the client process is doing.

Investigating client activity and application

In the first step, it has to be determined to which application the CLIENT PID on CLIENT host belongs to. Typically, this would be done using OS tools, like the 'ps' command in linux/unix or the taskmanager on windows systems. If the process is an SAP workprocess (disp+work.exe or dw), then further be conducted using transaction SM50 on the relevant SAP application server. In case of an 'uncomitted write transaction', it might actually be the clier blocking the commits. The SAP workprocess might be in state 'sleep' or waiting for locks/semaphore, thus preventing the execution of the commit.

Tools for analyzing SAP workprocess activity:

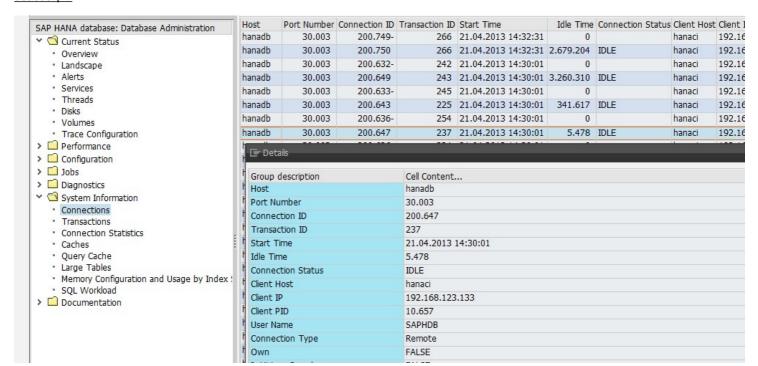
- Transactions SE30 or ST12
- In case of processes that appear to be hanging, call stacks can be generated into the developer traces (accessible using transaction ST11). Th commands to do so are 'kill -USR2 <pid>' (UNIX/LINUX) and 'sapntkill -USR2 <pid>' (Windows).

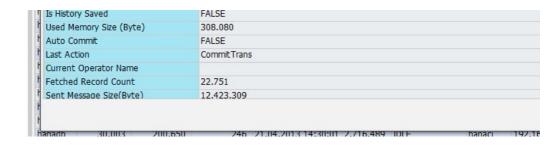
Analyzing database activity

Connections

If it can be ruled out that the problem is located on client side, the activity on HANA side has to be analyzed next. As an entry point, the CONNECTION used.

dbacockpit:





HANA Studio:

lame ▼	Description	
Used memory by tables	Shows total memory consumption of all column and row tables	
Transactions	Shows a list of transactions	
Table locks	Shows table locks	
Size of tables on disk	Shows the size of tables on disk in bytes	
Sessions	Shows details about sessions and their resource consumption	
Session context	Shows session context information	
Schema size of loaded tables	Shows memory consumption of schemas (loaded tables) in MB	
Record locks	Shows record locks	
Overall workload	Shows current workload	
Open transactions	Shows a list of open transactions	
Merge statistics	Shows merge statistics	
MVCC blocker transaction	Shows transaction which is blocking the garbage collection	
MVCC blocker connection	Shows connection which is blocking the garbage collection	
Lock waiting history	Shows summary of occurred lock waits	
HANA usage	Show usage information of the HANA system	
Database information	Basic configuration of the database	
Connections	Shows a list of connections	
Connection statistics	Shows connection statistics including network I/O	
	Shows connection attempts and status	
Component memory usage	Shows memory consumption of components	
Caches	Shows caches	
Blocked transactions	Shows a list of transactions waiting for a record lock	
Backup catalog	Backup catalog - Shows most recent backups and recoveries	

SQL Statement:

select * from m_connections where connection_id = <id>

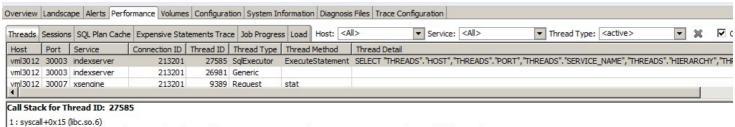
Threads

Since HANA is using multiple threads to process incoming request, an overview of all threads associated with a particular connection has to be create select * from m_service_threads where hierarchy like '<connection id>/%'

Tools to analyze HANA thread activity

Call stacks

The HANA Studio provides the functionality to display the call stacks of the HANA threads:



- 2: Synchronization::BinarySemaphore::timedWait(unsigned long, Execution::Context8)+0x265 at LinuxFutexOps.hpp:53 (libhdbbasis.so)
 3: TrexSync::Event::waitFor(int)+0x35 at TrexSync.cpp:367 (libhdbbasement.so)
 4: TrexThreads::Thread::join(int)+0x28 at Thread.cpp:204 (libhdbbasement.so)

While this information is mainly for SAP internal purposes, the following function names being visible in the call stack could indicate known issues:

■ TRexAPI::LateMatColumn: When observed during a seemingly hanging DSO migration, this function is an indication for the problem outlined in 1821785.

Header Data

Released On 14.05.2013 14:29:11 Release Status Released to Customer

Component BC-DB-HDB SAP HANA database

Priority Normal Category Problem

Product

This document is not restricted to a product or product version

References

This document refers to:

CSS SAP Notes

1821785 DSO conversion is hanging due low number of latematcol pages