Troubleshooting HANA Performance issues

Purpose:

The purpose of this page is to help you to determine the root cause of a performance issue on SAP HANA Database.

Need to first determine what type of performance issue you face:

Is one transaction/Report impacted or is it a system wide performance issue?

If one specific transaction or Report you generally need the following information for the problem analysis:

1: Recreate the problem for tracing purposes. Step by step description (with screenshots if possible) of how exactly the problem can be recreated may be required by SAP Support if you are not able to resolve the problem on your own.

2: If ABAP system an ST12 trace of the problem transaction/Report is required for analysis. This can be collected as described in the note ST12 “ABAP Trace for SAP EarlyWatch/GoingLive”. SAP Support would also require an R/3 support connection so that the ST12 trace can be analysed in the system. If after analysing ST12 trace (see information below) it is determined that the runtime is lost on the HANA Database then a plan viz trace (if the problem SQL statement can be found from the ST12 trace) or a SAP HANA Performance trace may be required for deeper analysis on the HANA Database side. For information on collecting these traces see 3 below.

3: If the SQL statement with the performance problem is already known then it can be helpful to have an executed Plan viz trace as described in note Create & Export PlanViz in HANA Studio and a SAP HANA Performance trace as per the KBA FAQ: SAP HANA Performance Trace. To generate a Plan Viz trace for a BW query from RSRT transaction see the documentation on BWonHana: PlanViz Trace generated in RSRT. On certain HANA Revisions there can be issues with collecting the plan viz trace, the impacted HANA Revisions are documented in the KBA How-To: Configuring SAP HANA Traces. For the SAP HANA Performance trace to reduce the size of the .tpt file that is generated and to only collect the SAP HANA Database traffic that is related to the performance issue you should specify the 'application user' when enabling the trace and normally you want to flag the options for execution plan and function profiler when collecting the trace. You should not let the SAP HANA Performance trace run for longer than 30 minutes.

If system wide performance problem:

As a first step execute the HANA Mini check report (HANA_Configuration_Minichecks) from the KBA 1969700 - SQL Statement Collection for SAP HANA, with the configuration you get an overview of the overall health of the system in terms of Memory, CPU, I/O, locks etc and other potential issues that may cause a performance problem on the system. If there is a potentially critical issue found with the configuration check script the column ‘C’ is marked with an ‘X’ and the last column of the report (SAP_NOTE) references the relevant SAP note or KBA that you need to follow to resolve the issues(s) found by the script.

Analysis:

For ST12 trace:

If you want more details on how to collect ST12 trace and how to analyse please see the KBA Step by step instructions on how to use ST12 trace for analysis.

From HANA Database point of view in the ABAP performance trace we are interested in % of the trace that is ABAP, DB etc and the ABAP Calls(s) that take most of the time sorted descending by Net(%). In general if most of the time is ABAP then it is often not a database problem but there is an issue with the standard ABAP coding or the custom code.

Known issues and how to resolve:

1: System is slow in general due to problem with CPU, MEMORY. I/O or table locks, use HANA_Configuration_Minichecks script to check and follow recommended notes depending on issue found with configuration script.

2: After migration to HANA DB for ECC or Suite on HANA system reports and transactions are slow, check that you are using HANA optimized transactions and application switches are enabled as per the PDF called SoH_Optimizations_2014_10_24.pdf attached to the note SAP ERP powered by SAP HANA - Optimizations.

3: After migration to HANA DB custom transactions or ABAP programs are slow:

If most of the time is ABAP but in custom program or transaction then you need to optimize this using code inspector tool and SQL monitor, see the note ABAP custom code migration for SAP HANA - recommendations and Code Inspector variants for SAP HANA migration- ABAP custom code migration for SAP HANA.

This older post also have useful information especially the part on The Golden Rules: https://ABAP_on_HANA – from analysis to optimization/

4: SELECT TOP X type statements cause performance problem on HANA (symptom : system is slow with a lot of these statements running), see the note High CPU Consumption Caused by UnifiedTable::MVCCObject::generateOLAPBitmapMVCC.

5: Poor performance of calculation view(s) after HANA Upgrade, see the following notes:
High query compilation times and absence of plan cache entries for queries against calculation views

SAP HANA DB: Disable/Enable CalculationEngine Feature - CalcView Unfolding

6: Poor performance of queries due to FDA (Fast data access), see question 6 from the KBA FAQ: SAP HANA Fast Data Access (FDA)

7: Due to issue with memory management of FDA queries HANA System has poor performance and you see entries like ‘Destroying allocator ‘Connection/xxxxx/Pool/RowEngine/Session’ with x blocks and xxx byte still allocated’ in the HANA Trace files, see KBA Memory Leak in Pool/RowEngine/Session

**Using the Explain Plan to optimize queries on HANA:**

The explain plan allows you to evaluate the execution plan that the SAP HANA Database uses to execute an SQL statement.

The Explain can be used as an alternative to the Plan visualization trace if you are on a SAP HANA Database revision where there are known issues with the Plan visualization trace as described in the details part for the ‘section PlanViz / Execution trace’ in the KBA 2119087 - How-To: Configuring SAP HANA Traces

**How to Collect the Explain plan:**

The explain Plan can be collected in several different ways. From HANA Studio, you can right mouse click on the statement and then select the explain plan option:

![HANA Studio Explain Plan](image)

The following SQL statements can also be used to collect the explain plan:

EXPLAIN PLAN [SET STATEMENT_NAME = "<statement_name>"] FOR SELECT <subquery>*

EXPLAIN PLAN [SET STATEMENT_NAME = "<statement_name>"] FOR SQL PLAN CACHE ENTRY <sql_plan_id>*

**Parameters:**

<statement_name> ::= string literal used to identify the name of a specific execution plan in the output table for a given SQL statement. It is set
to NULL if the SET STATEMENT_NAME is not specified.

\[\text{subquery} ::= \text{an SQL statement}\]

\[\text{sql\_plan\_id} ::= \text{an SQL plan id (in sql plan cache)}\]

The Explain plan can also be collected from transaction DBACOCKPIT on the application server:

From transaction DBACOCKPIT you will only see partial information on the explain plan, in the output you will see the Operations executed and order they were executed in but not the information on cost and table size estimations (OUTPUT_SIZE and SUBTREE_COST, see table Columns in EXPLAIN_PLAN_TABLE view). It is possible to get this missing information by taking the executed statement from the plan cache and executing the statement again.

Additional information on the Explain Plan collection is available in SAP note 2410208 - Collect Explain Plan of a Prepared Statement

**Where to find information on the explain plans:**

You can obtain the SQL statements and the associated SQL explain plans from EXPLAIN_PLAN_TABLE view. The view is shared by all users. The table below describes the meaning of the columns in EXPLAIN_PLAN_TABLE view:

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATEMENT_NAME</td>
<td>The string specified as STATEMENT_NAME on executing the EXPLAIN PLAN command. This is used to distinguish plans from each other when there are multiple plans in the EXPLAIN_PLAN_TABLE view.</td>
</tr>
</tbody>
</table>
How to use the Explain Plan to optimize a query:

The explain plan can be used to optimize a query, to reduce the run time of the query or reduce the memory consumption of the query. From the execution plan you can find a lot of important information on the query execution and the different operations involved in the background. From the plan you can see what type of operation was executed on HANA (Joins, Unions, aggregations etc), what engine was used on HANA (ROW, COLUMN, OLAP, HEX, ESX), what tables/objects were accessed, were Indexes/concats used, what tasks can be executed in parallel and what tasks need to be executed in sequence, estimation of cost of operation etc. For a detailed explanation of the information that can be obtained from explain plan please see the information in table Columns in EXPLAIN_PLAN_TABLE view above.

From the partial query explain plan below we can see the OUTPUT_SIZE and SUBTREE_COST estimations for the various operations. We can see that for the operations 1-13 the ROW store engine is used and that the operations 1-5 are executed sequentially and the operations 6-11 can be executed in parallel. Whether the operations can be executed in sequence or parallel can be seen from the indentation for the operation in the column OPERATOR_NAME:
In the above case as an example optimization it may be possible to change the query operations so that the Column Engine is used instead of ROW store engine on HANA if a similar operator exists for the column store engine. From the explain plan you may find that the most optimal HANA Engine is not used, in this case there are hints available for the SAP HANA Database that you can use to influence how a query is executed on HANA. For example, the hint use_olap_plan will force the HANA Database to use the OLAP engine instead of the join engine where this is technically possible. The available hints for the SAP HANA Database are described in the Knowledge base article 2142945 - FAQ: SAP HANA Hints. It may also be possible to adjust the above query so that more steps can be executed in parallel.

Later in the example explain plan output there is a HANA Engine switch between column and row store:

Where possible an Engine switch on HANA should be avoided as it requires a Materialisation of intermediate results and this is expensive in terms of both query performance and memory usage on HANA. A further optimization in this case would be adjust the query to avoid engine switch if possible.

You can optimize the performance of a query in terms of speed and memory usage using the output of the explain plan and the information available in the KBA 2000002 - FAQ: SAP HANA SQL Optimization

TOP Notes/KBA's for Performance analysis on HANA:

2000000 - FAQ: SAP HANA Performance Optimization
2000002 - FAQ: SAP HANA SQL Optimization
2222217 - How-To: Troubleshooting SAP HANA Startup Times

Guided answers for Performance analysis on HANA:

Master Guided Answer : Performance issues
How to perform a Network analysis
Analysis of Hardware Bottlenecks for Performance
Performance Analysis Procedure for an ABAP Program