Aspects of the HANA-optimized InfoCube

Contents

Aspects of the HANA-optimized InfoCube 1

Author 1

Introduction – From relational to in-memory thinking 1

HANA-Optimized InfoCubes – Simplified Modeling, Faster Loads 1

HANA-Optimized InfoCubes – How to Get It 3

HANA-Optimized InfoCubes – What do I have to consider? 3

HANA-Optimized InfoCubes and Data Store Objects (DSOs) 4

Take-aways 6

References 6

Author

Klaus Nagel: Development Manager, TIP In-Memory Platform BW (SAP AG).

Klaus Nagel joined SAP more than 10 years ago and started as a Developer in the BW team. He now holds the position of Development Manager and leads the BW&HANA Data Management team.

Introduction – From relational to in-memory thinking

Multi-dimensional models for analytic applications are represented on a classic RDBMS as star schemas. A valid star schema implementation is one of the most challenging tasks – a bad design has direct negative impact on data load and query performance. The question arises whether the strong link between logical multidimensional model and the data base implementation as a star schema does still hold in times of a modern in-memory database like SAP HANA.

HANA-Optimized InfoCubes – Simplified Modeling, Faster Loads

The InfoCube in BW is represented by the **BW-extended star schema** with 2 fact tables (E-table with read-optimized partitioning, F-table with write/delete-optimized partitioning), dimension tables as grouping sets of characteristics and shared Masterdata tables (see figure 1 – left side). While this schema is optimized for classic RDBMS technology, it is not required for a modern in-memory database like SAP HANA. The schema can be greatly simplified by using only one fact table (HANA can perform read, write and delete operations equally fast on the same layout) and joining directly to the Masterdata tables (see figure 1 – right side).

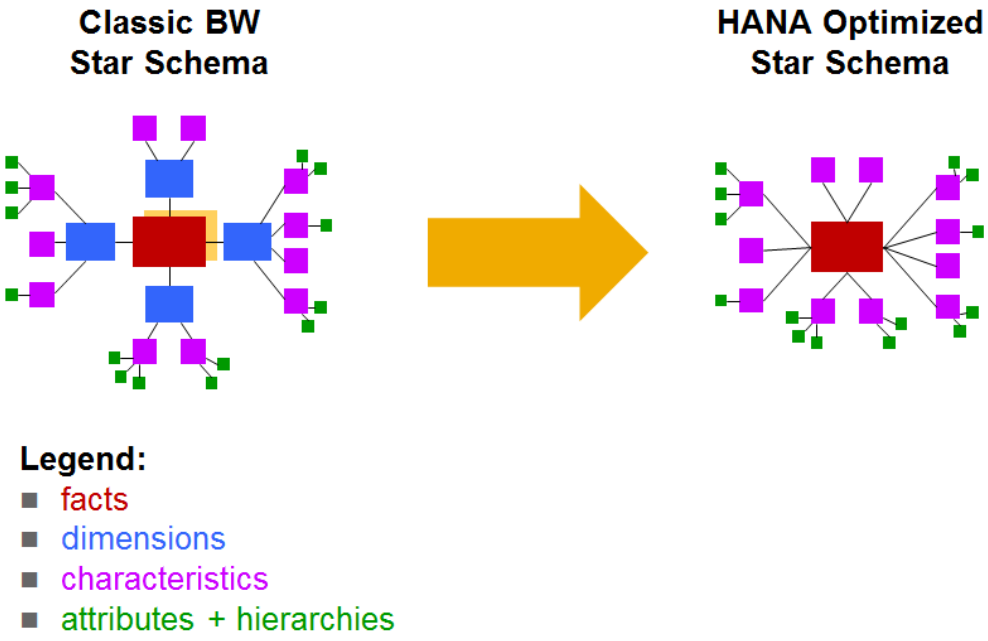


Figure 1: Classic BW-extended Star Schema versus HANA-optimized Star Schema

The main benefits of such a simplified structure are

- **Simplified modeling**: Dimensions as grouping sets of characteristics are still available in the InfoCube modeling, but they are pure metadata now. They do not impact the physical representation on the database any longer, such eliminating the need to take this aspect into account when modeling an InfoCube. I.e. there will be no more "badly" modeled InfoCubes with huge dimension tables and its corresponding negative impact on querying and loading performance. Note that there are also no more Line Item dimensions (all characteristics are physically Line Items now) and there is no need to set the high cardinality flag anymore.

- **Simplified re-modeling**: Adding a new keyfigure to an InfoCube in BW-on-HANA is easy, since adding a column on a columnar storage is a very fast operation. But now it is also easy to move characteristics from one dimension to another dimension. This is a pure metadata change and can be done without re-modeling, but directly in the InfoCube maintenance.

- **Faster loading**: Creating the BW-extended star schema during InfoCube load can take most of the time depending on the number of characteristics and the size of the dimension tables. The DIMIDs that represent the foreign keys to join fact tables and dimensions tables need to be looked up for new tuples or have to be created if they do not yet exists. On an average we have seen in real customer examples that loading into a HANA-optimized InfoCube is 3-5 times faster than into the Standard InfoCubes.

The Query performance does not improve in general due to the HANA-optimized schema. You can find queries that benefit from the classic schema as well as queries that improve on the new schema. But the runtime difference is negligible.

Note that there is one exception with respect to the table layout. We still have a package dimension table for the HANA-optimized InfoCube. Here the advantage to do very fast Request deletion is simply too big, while the effect on load performance to create the package dimension entries is negligible.

HANA-Optimized InfoCubes – How to Get It

To get to a HANA-optimized InfoCube is a really simple **conversion** step. You can either trigger it from the InfoCube maintenance (Menu: GoTo → "Conversion to SAP HANA optimized …") or via program RSDRI_CONVERT_CUBE_TO_INMEMORY. You can select one or many InfoCubes. The application log is either shown directly or can be found using the filter "OBJECT = RSDRI, SUBOBJECT = REFACTOPRING" in transaction SLG1.

After the conversion the HANA-optimized InfoCube behaves just as the Standard InfoCube before, i.e. there is no need to change any load process or DTP, nor MultiProviders or Queries on top!

During a conversion a lock is set, preventing all maintenance activity and load processes. Querying on the InfoCubes is however possible during this time.

Conversion times: The conversion is executed as a Stored Procedure within HANA and therefore shows an excellent performance. We have tested the conversion routine with many customer scenarios and have seen always runtimes of only a few minutes even for very large InfoCubes (e.g. 12 minutes for an InfoCube with more than 400 million records). The exact runtime depends of course on the distribution of the data, number of fields, especially characteristics, and the hardware of the HANA appliance.

HANA-Optimized InfoCubes – What do I have to consider?

Currently there are **no limitations** known for HANA-optimized InfoCubes, also InfoCubes with non-cumulative keyfigures can be converted (also if data is loaded via a "3.x dataflow" (Update rules) – we, of course, strongly recommend to switch to the new dataflow (DTP) logic, as they benefit a lot more from HANA than the old ones!).

We therefore recommend to convert all InfoCubes to the new HANA-optimized type, but this can be done step-by-step and the conversion can run in load windows over night or on weekends.

As for InfoCubes in a classic BW system, a maximum of 233 keyfigures, 16 dimensions, and 248 characteristics can be modeled for a HANA-optimized InfoCube.

Partitioning: The fact table of a HANA-optimized InfoCube can (and must) no longer be partitioned semantically, i.e. using a time characteristic to split the data into disjoint partitions. Query performance is fast even without such partitions on very large fact tables. However, overall 4 different partitions are created automatically for every fact table: Partition 1 for non-compressed Requests, Partition 2 for compressed Requests, Partition 3 for reference points of the inventory data, and Partition 4 for so-called historic movements of inventory data. Partitions 3 and 4 are created also if the InfoCube does not contain any inventory/non-cumulative keyfigure, they are then always empty.

Even so there are no two different fact tables anymore for HANA-optimized InfoCubes, you still have the option to run **InfoCube-compression**, i.e. aggregate an interval of Requests into Request = 0 in the InfoCube. This can be especially helpful, if you expect a significant size reduction due to many inverse bookings of requests. But it is not required to run the compression for query performance reasons, since the impact of size difference before and after compression of the fact table is negligible for the read performance. Compressing the InfoCube can however have a positive impact on the load time, or to be exact the MERGE time for an InfoCube fact table (for details on the MERGE process s in BW-on-HANA see [1]). As mentioned above, fact data for Request > 0 and Request = 0 are stored in different partitions, therefore the MERGE process after loading a Request to an InfoCube only runs on the “Request > 0” partition – and the smaller this partition, the faster the MERGE. But this becomes only relevant for large and very large InfoCubes. “Request > 0” partitions with 10-50 million records are not a problem, i.e. if your InfoCube is only of this size, there is no need to run compression at all. InfoCube-compression is also executed as a HANA-optimized StoredProcedure.

Inventory handling: The handling of “historic movements” has changed for the new HANA-optimized InfoCubes (only those!). Previously you had to identify the requests with historic movement after they were loaded and had to run InfoCube-compression with a specific setting (“no marker update”) for those (if you forgot this or forgot to remove the setting afterwards, you had corrupt data in the InfoCube!). Now this setting is part of the DTP, i.e. you identify already in the load process, if the data are “historic movements”. These requests are then marked via “RECORDTP = 2” in the package dimension table and handled accordingly during querying and InfoCube-compression (RECORDTP = 0 are deltas, RECORDTP = 1 are reference points). The setting for “historic movements” for the InfoCube-compression is then no longer necessary (and available) in the InfoCube manage screen (checkbox “no marker update” is no longer displayed). Exception: If data is loaded to the HANA-optimized InfoCube via a 3.x dataflow the setting is still available and has to be set for the requests prior to compressions. In such cases the RECORDTP is updated to 2 prior to the actual compression algorithm.

Any new InfoCube created in a BW-on-HANA system will automatically be a HANA-optimized InfoCube. There is no possibility (and no need) to create Standard InfoCubes any more since the HANA-optimized covers all aspects of an InfoCube (even better). It does not matter whether the InfoCube is created in the InfoCube maintenance, via the BAPi or imported via transport.

We strongly recommend to have the complete BW landscape (Sandbox/Test → Quality → Production) on BW-on-HANA to have a clear test and consolidation path. However it may be that you need to transport an InfoCube from a non-HANA BW system into a BW-on-HANA system. This is possible and the InfoCube is automatically converted into a HANA-optimized InfoCube as part of the “After-Import” method.

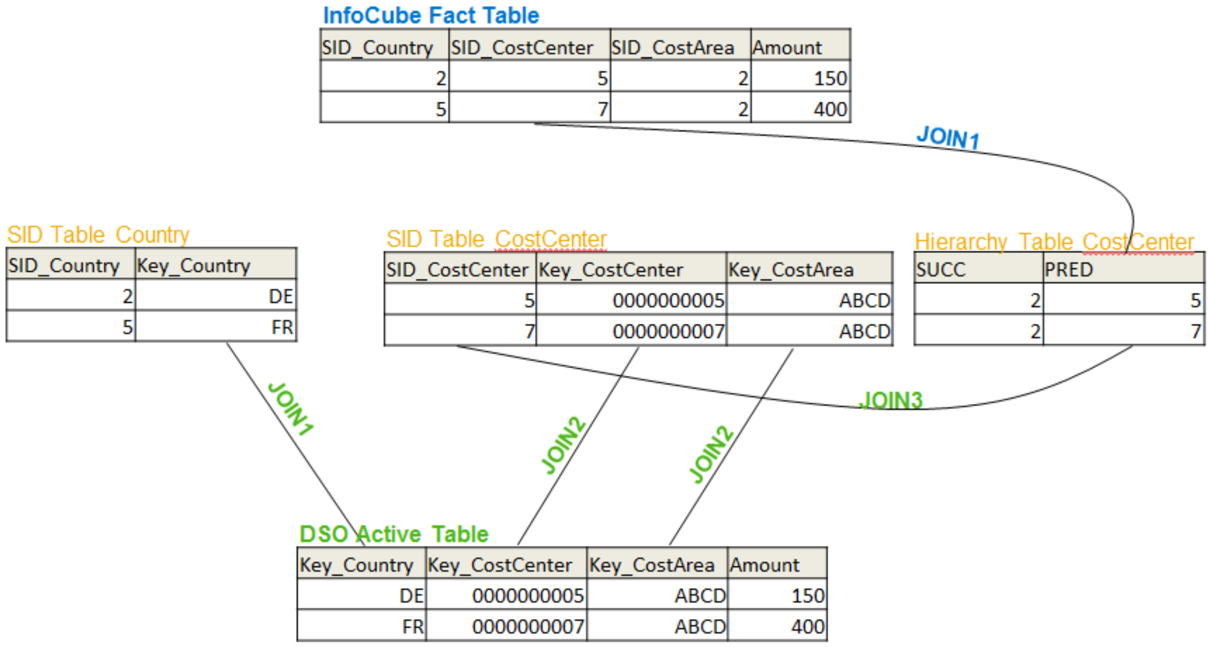
Note that for any type of InfoCube, not only the HANA-optimized ones, you do no longer need DB indexes and DB statistics anymore. Therefore you can remove all these processes within your process chains – which again leads to a faster and more simplified data load. If you do not remove these processes in the process chains, they will simply do nothing.

HANA-Optimized InfoCubes and Data Store Objects (DSOs)

Since all **DSOs** (not only the HANA-optimized DSOs) are based **on in-memory data and columnar tables** read access to DSOs is much, much faster compared to DSO access on traditional databases. But the real boost in query performance comes if we can push BW-OLAP operations down to the Calculation Engine of HANA (Hierarchy processing, MultiProvider UNION, restricted keyfigures, exception aggregation, currency conversion, and many more to come). The access to the HANA

Calculation Engine is then not via standard SQL, but via a special API that BW already leverages in the case of BW Accelerator. And it requires SIDs for all characteristic values. Therefore the optimized Calculation Engine access to DSO data requires that the DSO has the property "SID generated during activation" (we will publish more details on this and the huge performance improvements for the SID generation during DSO activation with HANA shortly here on SDN as well, currently see [2]).

So for DSOs with the "SID-flag" turned on, we can access the data via the Calculation Engine and we see in all customer examples and test cases so far **the same excellent performance** when **reporting on a DSO or an InfoCube** (of course only if both contain the same data in the same granularity). This is the case despite the fact that in general a query on a DSO has to process more JOINs than a query on an InfoCube (see figure 2).



InfoCube Fact Table

SID_Country	SID_CostCenter	SID_CostArea	Amount
2	5	2	150
5	7	2	400

SID Table Country

SID_Country	Key_Country
2	DE
5	FR

SID Table CostCenter

SID_CostCenter	Key_CostCenter	Key_CostArea
5	0000000005	ABCD
7	0000000007	ABCD

Hierarchy Table CostCenter

SUCC	PRED
2	5
2	7

DSO Active Table

Key_Country	Key_CostCenter	Key_CostArea	Amount
DE	0000000005	ABCD	150
FR	0000000007	ABCD	400

Figure 2: Statement for a Query with a GROUP BY on Country and a filter on a hierarchy node for a DSO and an InfoCube

So you might ask, **why do we still offer InfoCubes** at all and why do we not recommend to switch to DSOs only. Well, first of all InfoCubes are still there for the simple reason of the non-disruptive approach of BW-on-HANA. After the migration to HANA as database, the BW system behaves as before and any further optimizations with respect to the new HANA-based capabilities can be turned on step-by-step. But there are also some hard and some soft facts of why InfoCubes may still be required.

Hard facts (as of now):

- non-cumulative keyfigures can only be modeled in InfoCubes,
- integrated planning can only be done on (transactional) InfoCubes,
- the external write-interface (RSDRI) only works for InfoCubes .

Soft facts:

- an InfoCube represents a multi-dimensional model – a DSO not. If you use, as recommended anyway, a MultiProvider as abstraction layer on top, this aspect can be ignored
- the Update handling of InfoCubes and DSOs differ significantly: An InfoCube is insert-only (or, equivalent, all characteristics are keys), while a DSO still has the limitation of maximal 16 keys. This has a different semantic and also has an impact on e.g. auditing that has to be taken into account.
- data visibility: in a non-transactional InfoCube data is only visible after the complete request has been successfully loaded, while data becomes visible in a DSO during activation based on the Commits after packages.

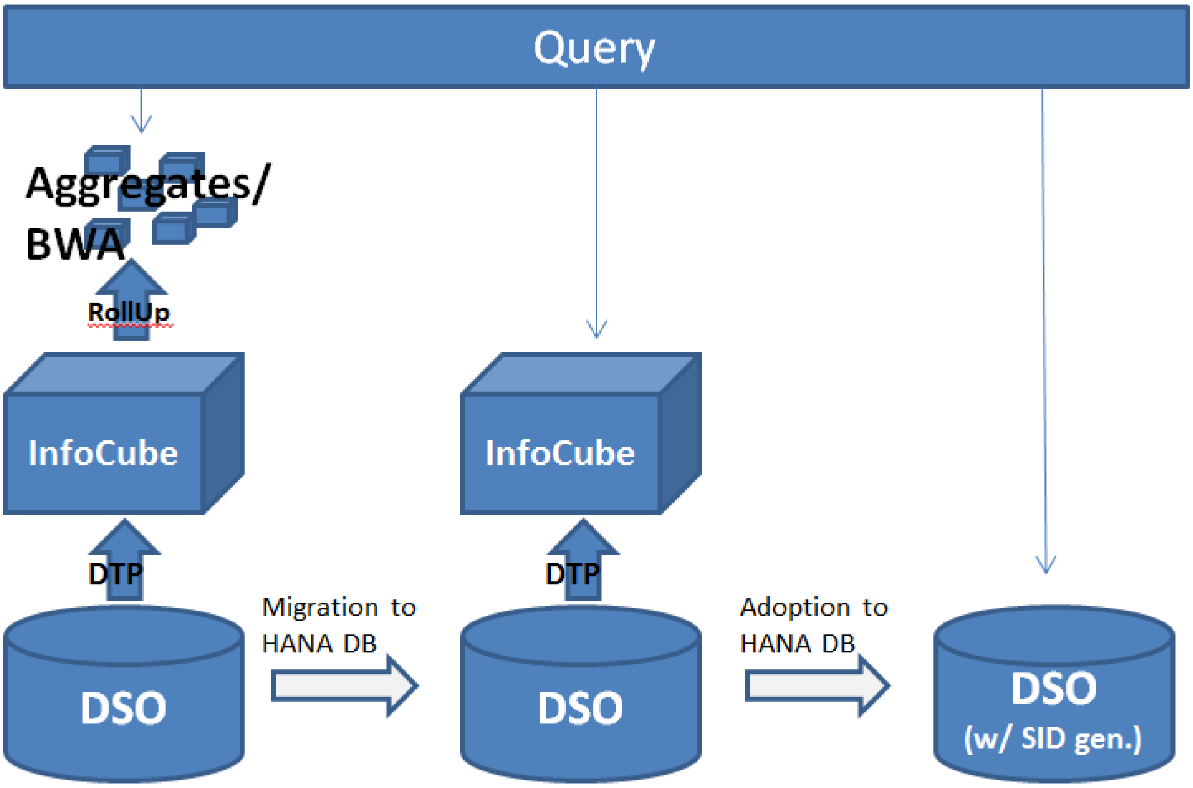


Figure 3: Possible reduction of layers with the migration and adoption to HANA.

Take-aways

- Conversion to HANA-optimized InfoCubes is simple, fast and brings big benefits – and you can do this step-by-step
- Removing the InfoCube-layer for certain scenarios and reporting directly on DSOs is possible and removes another materialized layer – but it has to be assessed scenario by scenario based on Query performance and semantic/business requirements

References

[1] SDN Article on HANA-MERGE process in BW: to be published shortly

[2] SAP Online Help, HANA-optimized DSO: http://help.sap.com/saphelp_nw73/helpdata/de/32/5e81c7f25e44abb6053251c0c763f7/content.htm