SAP HANA Database - Data Management for Modern Business Applications

Franz Färber [#1], Sang Kyun Cha [+2], Jürgen Primsch [?3], Christof Bornhövd [*4], Stefan Sigg [#5], Wolfgang Lehner [#6]

[#] *SAP – Dietmar-Hopp-Allee 16 – 69190, Walldorf, Germany*
[+] *SAP – 63-7 Banpo 4-dong, Seochoku – 137-804, Seoul, Korea*
[?] *SAP – Rosenthaler Str. 30 – 10178, Berlin, Germany*
[*] *SAP – 3412 Hillview Ave – Palo Alto, CA 94304, USA*

[1] franz.faerber@sap.com [2] sang.k.cha@sap.com [3] j.primsch@sap.com
[4] christof.bornhoevd@sap.com [5] stefan.sigg@sap.com [6] wolfgang.lehner@sap.com

ABSTRACT

The SAP HANA database is positioned as the core of the SAP HANA Appliance to support complex business analytical processes in combination with transactionally consistent operational workloads. Within this paper, we outline the basic characteristics of the SAP HANA database, emphasizing the distinctive features that differentiate the SAP HANA database from other classical relational database management systems. On the technical side, the SAP HANA database consists of multiple data processing engines with a distributed query processing environment to provide the full spectrum of data processing – from classical relational data supporting both row- and column-oriented physical representations in a hybrid engine, to graph and text processing for semi- and unstructured data management within the same system.

From a more application-oriented perspective, we outline the specific support provided by the SAP HANA database of multiple domain-specific languages with a built-in set of natively implemented business functions. SQL – as the lingua franca for relational database systems – can no longer be considered to meet all requirements of modern applications, which demand the tight interaction with the data management layer. Therefore, the SAP HANA database permits the exchange of application semantics with the underlying data management platform that can be exploited to increase query expressiveness and to reduce the number of individual application-to-database round trips.

1. INTRODUCTION

Data management requirements for enterprise applications have changed significantly in the past few years. For example, it is no longer reasonable to continue the classical distinction between transactional and analytical access patterns. From a business perspective, queries in transactional environments on the one hand are building the sums of already-delivered orders, or calculating the overall liabilities per customer. On the other hand, analytical queries require the immediate availability of operational data to enable accurate insights and real-time decision making. Furthermore, applications demand a holistic, consistent, and detailed view of its underlying business processes, thus leading to huge data volumes that have to be kept online, ready for querying and analytics. Moreover, non-standard applications like planning or simulations require a flexible and graph-based data model, e.g., to compute the maximum throughput of typical business relationship patterns within a partner network. Finally, text retrieval technology is a must-have in state-of-the-art data management platforms to link unstructured or semi-structured data or results of information retrieval queries to structured business-related contents.

In a nutshell, the spectrum of required application support is tremendously heterogeneous and exhibits a huge variety of interaction patterns. Since classical SQL-based data management engines are too narrow for these application requirements, the SAP HANA database presents itself as a first step towards a holistic data management platform providing robust and efficient data management services for the specific needs of modern business applications [5].

The SAP HANA database is a component of the overall SAP HANA Appliance that provides the data management foundation for renovated and newly developed SAP applications (see Section 4). Figure 1 outlines the different components of the SAP HANA Appliance. The SAP HANA Appliance comprises replication and data transformation services to easily move SAP and non-SAP data into the HANA system, modeling services to create the business models that can be deployed and leveraged during runtime, and the SAP

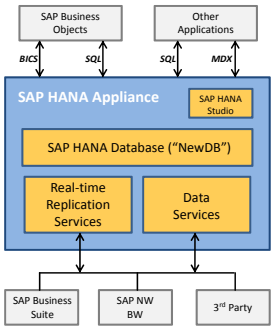


Figure 1: Components of the SAP HANA Appliance

HANA database as its core. For the rest of this paper, we specifically focus on the SAP HANA database.

Core Distinctive Features of the SAP HANA Database

Before diving into the details, we will outline some general distinctive features and design guidelines to show the key differentiators with respect to common relational, SQL-based database management systems. We believe that these features represent the cornerstones of the philosophy behind the SAP HANA database:

- *Multi-engine query processing environment*: In order to cope with the requirements of managing enterprise data with different characteristics in different ways, the SAP HANA database comprises a multi-engine query processing environment. In order to support the core features of enterprise applications, the SAP HANA database provides SQL-based access to relationally structured data with full transactional support. Since more and more applications require the enrichment of classically structured data with semi-structured, unstructured, or text data, the SAP HANA database provides a text search engine in addition to its classical relational query engine. The HANA database engine supports "joining" semi-structured data to relations in the classical model, in addition to supporting direct entity extraction procedures on semi-structured data. Finally, a graph engine natively provides the capability to run graph algorithms on networks of data entities to support business applications like production planning, supply chain optimization, or social network analyses. Section 2 will outline some of the details.

- *Representation of application-specific business objects*: In contrast to classical relational databases, the SAP HANA database is able to provide a deep understanding of the business objects used in the application layer. The SAP HANA database makes it possible to register "semantic models" inside the database engine to push down more application semantics into the data management layer. In addition to registering semantically richer data structures (e.g., OLAP cubes with measures and dimensions), SAP HANA also provides access to specific business logics implemented directly deep inside the database engine. The SAP HANA Business Function Library encapsulates those application procedures. Section 3 will explain this feature from different perspectives.

- *Exploitation of current hardware developments*: Modern data management systems must consider current developments with respect to large amounts of available main memory, the number of cores per node, cluster configurations, and SSD/flash storage characteristics in order to efficiently leverage modern hardware resources and to guarantee good query performance. The SAP HANA database is built from the ground up to execute in parallel and main-memory-centric environments. In particular, providing scalable parallelism is the overall design criteria for both system-level up to application-level algorithms [6, 7].

- *Efficient communication with the application layer*: In addition to running generic application modules inside the database, the system is required to communicate efficiently with the application layer. To meet this requirement, plans within SAP HANA development are, on the one hand, to provide shared-memory communication with SAP proprietary application servers and more closely align the data types used within each. On the other hand, we plan to integrate novel application server technology directly into the SAP HANA database cluster infrastructure to enable interweaved execution of application logic and database management functionality.

2. ARCHITECTURE OVERVIEW

The SAP HANA database is a memory-centric data management system that leverages the capabilities of modern hardware, especially very large amounts of main memory, multi-core CPUs, and SDD storage, in order to improve the performance of analytical and transactional applications. The HANA database provides the high-performance data storage and processing engine within the HANA Appliance.

Figure 2 shows the architecture of the HANA database system. The Connection and Session Management component creates and manages sessions and connections for the database clients. Once a session has been established, database clients can use SQL (via

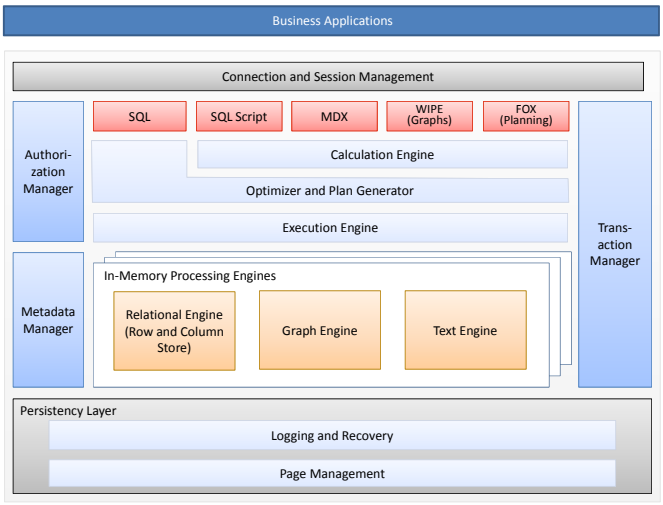


Figure 2: The SAP HANA database architecture

JDBC or ODBC), SQL Script, MDX or other domain-specific languages like SAP's proprietary language FOX for planning applications, or WIPE, which combines graph traversal and manipulation with BI-like data aggregation to communicate with the HANA database. SQL Script is a powerful scripting language to describe application-specific calculations inside the database.

SQL Script is based on side-effect-free functions that operate on database tables using SQL queries, and it has been designed to enable optimization and parallelization.

As outlined in our introduction, the SAP HANA database provides full ACID transactions. The Transaction Manager coordinates database transactions, controls transactional isolation, and keeps track of running and closed transactions. For concurrency control, the SAP HANA database implements the classical MVCC principle that allows long-running read transactions without blocking update transactions. MVCC, in combination with a time-travel mechanism, allows temporal queries inside the Relational Engine.

Client requests are parsed and optimized in the Optimizer and Plan Generator layer. Based on the optimized execution plan, the Execution Engine invokes the different In-Memory Processing Engines and routes intermediate results between consecutive execution steps.

SQL Script and supported domain-specific languages are translated by their specific compilers into an internal representation called the "Calculation Model". The execution of these calculation models is performed by the Calculation Engine. The use of calculation models facilitates the combination of data stored in different In-Memory Storage Engines as well as the easy implementation of application-specific operators in the database engine.

The Authorization Manager is invoked by other HANA database components to check whether a user has the required privileges to execute the requested operations. A privilege grants the right to perform a specified operation (such as create, update, select, or execute). The database also supports analytical privileges that represent filters or hierarchy drill-down limitations for analytical queries as well as control access to values with a certain combination of dimension attributes. Users are either authenticated by the database itself, or the authentication is delegated to an external authentication provider, such as an LDAP directory.

Metadata in the HANA database, such as table definitions, views, indexes, and the definition of SQL Script functions, are managed by the Metadata Manager. Such metadata of different types is stored in one common catalogue for all underlying storage engines.

The center of Figure 2 shows the three In-Memory Storage Engines of the HANA database, i.e., the Relational Engine, the Graph Engine, and the Text Engine. The Relational Engine supports both row- and column-oriented physical representations of relational tables. The Relational Engine combines SAP's P*Time database engine and SAP's TREX engine currently being marketed as SAP BWA to accelerate BI queries in the context of SAP BW. Column-oriented data is stored in a highly compressed format in order to improve the efficiency of memory resource usage and to speed up the data transfer from storage to memory or from memory to CPU. A system administrator specifies at definition time whether a new table is to be stored in a row- or in a column-oriented format. Row- and column-oriented database tables can be seamlessly combined into one SQL statement, and subsequently, tables can be moved from one representation form to the other [4]. As a

rule of thumb, user and application data is stored in a column-oriented format to benefit from the high compression rate and from the highly optimized access for selection and aggregation queries. Metadata or data with very few accesses is stored in a row-oriented format.

The Graph Engine supports the efficient representation and processing of data graphs with a flexible typing system. A new dedicated storage structure and a set of optimized base operations are introduced to enable efficient graph operations via the domain-specific WIPE query and manipulation language. The Graph Engine is positioned to optimally support resource planning applications with huge numbers of individual resources and complex mash-up interdependencies. The flexible type system additionally supports the efficient execution of transformation processes, like data cleansing steps in data-warehouse scenarios, to adjust the types of the individual data entries, and it enables the ad-hoc integration of data from different sources.

The Text Engine provides text indexing and search capabilities, such as exact search for words and phrases, fuzzy search (which tolerates typing errors), and linguistic search (which finds variations of words based on linguistic rules). In addition, search results can be ranked and federated search capabilities support searching across multiple tables and views. This functionality is available to applications via specific SQL extensions. For text analyses, a separate Preprocessor Server is used that leverages SAP's Text Analysis library.

The Persistency Layer, illustrated at the bottom of Figure 2, is responsible for the durability and atomicity of transactions. It manages data and log volumes on disk and provides interfaces for writing and reading data that are leveraged by all storage engines. This layer is based on the proven persistency layer of MaxDB, SAP's commercialized disk-centric relational database. The persistency layer ensures that the database is restored to the most recent committed state after a restart and that transactions are either completely executed or completely undone. To achieve this efficiently, it uses a combination of write-ahead logs, shadow paging, and savepoints.

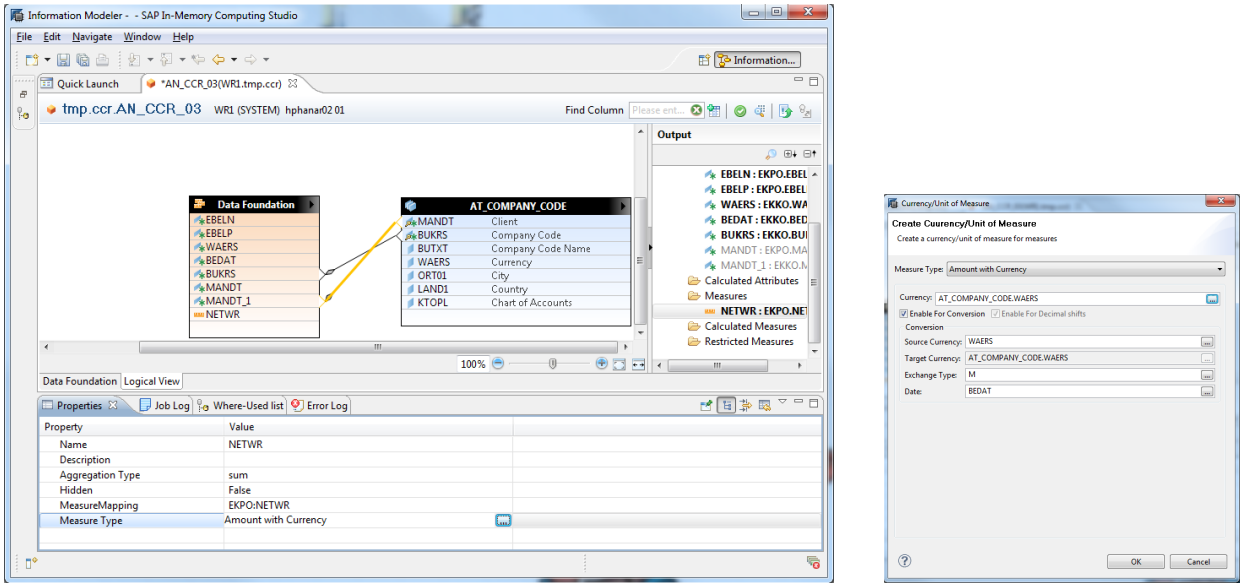
To enable scalability in terms of data volumes and the number of application requests, the SAP HANA database supports scale-up and scale-out. For scale-up scalability, all algorithms and data structures are designed to work on large multi-core architectures especially focusing on cache-aware data structures and code fragments. For scale-out scalability, the SAP HANA database is designed to run on a cluster of individual machines allowing the distribution of data and query processing across multiple nodes. The scalability features of the SAP HANA database are heavily based on the proven technology of the SAP BWA product.

3. SAP HANA DATABASE: BEYOND SQL

As outlined in our introduction, the SAP HANA database is positioned as a modern data management and processing layer to support complex enterprise-scale applications and data-intensive business processes. In addition to all optimizations and enhancements at the technical layer (modern hardware exploitation, columnar and row-oriented storage, support for text and irregularly structured data, etc.), the core benefit of the system is its ability to understand and directly work with business objects stored inside the database. Being able to exploit the knowledge of complex-structured business objects and to perform highly SAP application-specific business logic steps deep inside the engine is an important differentiator of the SAP HANA database with respect to classical relational stores.

More specifically, the "Beyond SQL" features of the SAP HANA database are revealed in multiple ways. On a smaller scale, specific SQL extensions enable the exposure of the capabilities of the specific query processing engines. For example, an extension in the WHERE clause allows the expression of fuzzy search queries against the text engine. An explicit "session" concept supports the planning of processes and *What if?* analyses. Furthermore, SQL Script provides a flexible programming language environment as a combination of imperative and functional expressions of SQL snippets. The imperative part allows one to easily express data and control flow logic by using DDL, DML, and SQL-Query statements as well as imperative language constructs like loops and conditionals. Functional expressions, on the other hand, are used to express declarative logics for the efficient execution of data-intensive computations. Such logic is internally represented as data flows that can be executed in parallel. As a consequence, operations in a data flow graph must be free of side effects and must not change any global states, neither in the database nor in the application. This condition is enforced by allowing only a limited subset of language features to express the logic of the procedure.

On a larger scale, domain-specific languages can be supported by specific compilers to the same logical construct of a "Calculation Model" [2]. For example, MDX will be natively translated into the internal query processing structures by resolving complex dimensional expressions during the compile step as much as possible by consulting the registered business object structures stored in the metadata catalog. In contrast to classical BI application stacks, there is no need for an extra OLAP server to generate complex SQL statements. In addition, the database optimizer is not required to "guess" the semantics of the SQL statements in order to generate the best plan – the SAP HANA database can directly ex-



(a) SAP Information Modeler (b) Concurrency conversion dialog

Figure 3: Modeling of currency conversion within SAP HANA

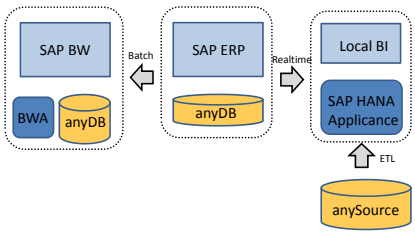
ploit the knowledge of the OLAP models carrying much more semantics compared to plain relational structures.

As an additional example of this "Beyond SQL" feature, consider the disaggregation step in financial planning processes [3]. In order to distribute coarse-grained planning figures to atomic entries—for example, from business unit level to department level—different distribution schemes have to be supported: relative to the actual values of the previous period, following constant distribution factors, etc. Since disaggregation is such a crucial operation in planning, the SAP HANA database provides a special operator, available within its domain-specific programming language, for planning scenarios. Obviously, such an operator is not directly accessible via SQL. Following this principle, the SAP HANA database also provides a connector framework to work with "external" language packages like the statistical programming environment R [1].

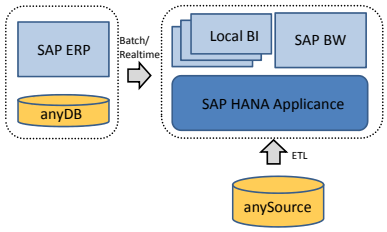
In addition to specifically tailored operators, the SAP HANA database also provides a built-in Business Function Library (BFL) that offers SAP-specific application code. All business logic modules are natively integrated into the database kernel with a maximum degree of parallelism. Compared to classical stored procedures or stored functions, the BFL is included in the database engine using all the technical advantages of deep integration. A prominent example of an application-specific algorithm is the procedure of currency conversion. Though supposedly simple in nature—a scalar multiplication of a monetary figure with the conversion rate—the actual implementation covering the complete application semantics of currency conversion comprises more than 1,000 lines of code. Figure 3(a) illustrates the graphical tool to create a "Calculation Model" in the SAP HANA database by applying a currency conversion function to an incoming data stream. As can be seen, the data source itself comprises not only simple columns but also comprehensive metadata such as type information with respect to plain, calculated, or derived measures.

The application designer creates a logical view using the Information Modeler and applies pre-defined application logics provided by the BFL. As shown in the modeling dialog of Figure 3(b), the parameters of the currency conversion function can be set in multiple ways to instrument the business logic. In the current example, the function performs a conversion to the currency with respect to the specific company code (given in column `AT_COMPANY_CODE.WAERS`).

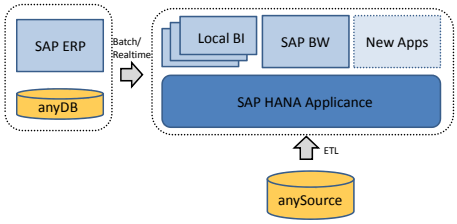
To summarize, the SAP HANA database provides a classical SQL interface including all transactional properties required from a classical database management system. In addition, the SAP HANA database positions itself as a system "Beyond SQL" by providing an ecosystem for domain-specific languages with particular internal support on the level of individual operators. Moreover, the concept of a BFL to provide a set of complex, performance-critical, and standardized application logic modules deep inside the database kernel creates clear benefits for SAP and customer-specific applications.



(a) Supporting local BI



(b) Running SAP BW



(c) Platform for new applications

Figure 4: Planned SAP HANA roadmap

4. THE HANA ROADMAP

Although, from a technology perspective, the SAP HANA database is based on the SAP BWA system with its outstanding record of successful installations, the generally novel approach of a highly distributed system with an understanding of semantic business models requires time for customers to fully leverage their data management infrastructure. SAP intends to pursue an evolutionary, step-wise approach to introduce the technology to the market.

In a first step, the SAP HANA Appliance is positioned to support local BI scenarios. During this step, customers can familiarize themselves with the technology exploiting the power of the new solution without taking any risk for existing mission-critical applications. SAP data of ERP systems will be replicated to the SAP HANA Appliance in real-time fashion. Data within the SAP HANA Appliance can be optionally enhanced by external non-SAP data sources and consumed using the SAP BOBJ analytical tools. Aside from new analytical applications on top of HANA, the primary use case here is the acceleration of operational reporting processes directly on top of ERP data.

The plan for the second phase of the roadmap, as shown in Figure 4(b), comprises supporting the full SAP BW application stack. Step by step, the customer is able to move more critical applications (like data warehousing) to the SAP HANA Appliance. This phase also positions the SAP HANA Appliance as the primary persistent storage layer for managed analytical data. Switching the data management platform will be a non-disruptive move from the application's point of view. In addition to providing the data management layer for a centralized data-warehouse infrastructure, the SAP HANA Appliance is also planned to be used to consolidate local BI data marts exploiting a built-in multi-tenancy feature.

The third step in the current roadmap – introducing the SAP HANA Appliance to the market in an evolutionary way – consists of extending the HANA ecosystem with new applications using the modeling and programming paradigm of the SAP HANA database in combination with application servers. Depending on the specific customer setup, long-term plans are to put HANA also under the classical SAP ERP software stack.

To summarize, the basic steps behind the HANA roadmap are designed to integrate with customers' SAP installations without disrupting existing software landscapes. Starting small with local BI installations, putting the complete BW stack on top of HANA in combination with a framework to consolidate local BI installations, is considered a cornerstone in the SAP HANA roadmap.

5. SUMMARY

Providing efficient solutions for enterprise-scale applications requires a robust and efficient data management and processing platform with specialized support for transaction, analytical, graph traversal, and text retrieval processing. Within the SAP HANA Appliance, the HANA database represents the first step towards a new generation of database systems designed specifically to provide answers to questions raised by demanding enterprise applications. The SAP HANA database, therefore, should not be compared to classical SQL or typical key-value, document-centric, or graph-based NoSQL databases. HANA is a flexible data storage, manipulation, and analysis platform, comprehensively exploiting current trends in hardware to achieve outstanding query performance and throughput at the same time. The different engines within the distributed data processing framework provide an adequate solution for different application requirements. In this paper, we outlined our overall idea of the SAP HANA database, sketched out its general architecture, and finally gave some examples to illustrate how an SAP HANA database positions itself "Beyond SQL" by natively supporting performance-critical application logics as an integral part of the database engine.

6. ACKNOWLEDGMENTS

We would like to express our sincere thanks to all of our NewDB colleagues for making the HANA story a reality. We also would like to thank Glenn Pauley, SIGMOD Record Editor for Industrial Perspectives, for his helpful comments.

7. REFERENCES

[1] P. Grosse, W. Lehner, T. Weichert, F. Färber, and W.-S. Li. Bridging two worlds with RICE. In *VLDB Conference*, 2011.

[2] B. Jaecksch, F. Färber, F. Rosenthal, and W. Lehner. Hybrid Data-Flow Graphs for Procedural Domain-Specific Query Languages. In *SSDBM Conference*, pages 577–578, 2011.

[3] B. Jaecksch, W. Lehner, and F. Färber. A plan for OLAP. In *EDBT conference*, pages 681–686, 2010.

[4] J. Krüger, M. Grund, C. Tinnefeld, H. Plattner, A. Zeier, and F. Faerber. Optimizing Write Performance for Read Optimized Databases. In *DASFAA Conference*, pages 291–305, 2010.

[5] H. Plattner and A. Zeier. *In-Memory Data Management: An Inflection Point for Enterprise Applications*. Springer, Berlin Heidelberg, 2011.

[6] J. Schaffner, B. Eckart, D. Jacobs, C. Schwarz, H. Plattner, and A. Zeier. Predicting In-Memory Database Performance for Automating Cluster Management Tasks. In *ICDE Conference*, pages 1264–1275, 2011.

[7] C. Weyerhäuser, T. Mindnich, F. Färber, and W. Lehner. Exploiting Graphic Card Processor Technology to Accelerate Data Mining Queries in SAP NetWeaver BIA. In *ICDM Workshops*, pages 506–515, 2008.