Oracle Database for SAP Roadmap

ORACLE DATABASE 12c FOR SAP

Oracle/SAP Support Status*
- 12.1.0.2: supported until 07/2022 (Ext.)
- 12.2.0.1: supported until 11/2020 (Prim.), 03/2022 (Lim.)

Supported Platforms
- Generic Servers (On-Premise)
- Oracle Engineered Systems
- Oracle Cloud (OCI ExaCS)

New Features/Options
- Oracle Database In-Memory
- Advanced Compression (ILM)
- Oracle Multitenant

Entry Notes
- SAP Note 1914631 (12.1)
- SAP Note 2470660 (12.2)

ORACLE DATABASE 18c FOR SAP

Oracle/SAP Support Status*
- supported until 06/2021 (Prim.)

Supported Platforms
- Generic Servers (On-Premise)
- Oracle Engineered Systems
- Oracle Cloud (OCI, ExaCS)

New Features/Options
- In-Memory Optimized Arithmetic
- Polymorphic Table Functions (ABAP® FOR ALL ENTRIES)

Entry Notes
- SAP Note 2705272
- SAP Note 260020

ORACLE DATABASE 19c FOR SAP

Oracle/SAP Support Status*
- supported until 04/2024 (Prim.), 04/2027 (Ext.)

Supported Platforms
- Generic Servers (On-Premise)
- Oracle Engineered Systems
- Oracle Cloud (OCI, ExaCS)

New Features/Options
- SQL Macros (for ABAP Core Data Services)
- Support of Microsoft Windows Server 2019, Oracle Linux 8, RHEL 8

Entry Notes
- SAP Note 2817074
- SAP Note 2799900

* Dates as of June 2020. For updates see MOS Note 742060.1 – (Prim.) = Primary Support, (Ext.) = Extended Support, (Lim.) = Limited Error Correction

ORACLE DATABASE FOR SAP: LATEST DATABASE TECHNOLOGY AND SUPPORT FOR APPLICATION OPTIMIZATIONS

Strategy and Roadmap

From the very beginning, the *Oracle Database for SAP or SAP on Oracle Database strategy* had been based on two pillars. The first pillar is the integration of Oracle Database features with the SAP environment. The second pillar is the integration of SAP application features with the Oracle database.

Today, both pillars supporting the SAP on Oracle Database strategy are clearly visible and important: Whenever Oracle releases a major new database feature, a development effort is needed to integrate it into the SAP architecture as well as the installation, administration and monitoring tools provided by SAP. Whenever SAP releases a new application optimization, a similar development effort is needed to integrate it with the Oracle Database technology.

The need to *integrate Oracle Database features with the SAP environment* has always been visible. It was particularly obvious, when Oracle released new database features for which the SAP architecture was not prepared. An example that many customers still remember is the project to integrate Real Application Clusters (RAC) into an SAP architecture based on the assumption that there can be many SAP Application Server instances, but only one Database Server instance. The certification of Oracle Multitenant was a similar architectural revolution and required no less effort than the RAC certification.

The need to *integrate SAP application features with the Oracle Database*, on the other hand, has only rarely been recognized. The classic SAP applications (such as R/3 and BW) were developed on the Oracle Database. Later on, when SAP started to support IBM DB2 and Microsoft SQL Server, they put the least common denominator strategy in place, i.e. they used only those database features that were available in all supported databases. Not much stress, therefore, on the Oracle Database.

This has changed with the advent of SAP's own database (HANA). SAP realized very soon that they had to drop the least common denominator strategy and change their applications: As long as SAP applications treat HANA as a database similar to all other databases, it is very difficult to convince customers that there is a benefit in implementing HANA. Therefore, SAP has embarked on an application optimization project in order to allow SAP applications to make use of special HANA features.

"Special HANA features", however, does not mean "HANA-only features". There is nothing in HANA that cannot be done by the Oracle Database as well. Therefore, the need to integrate SAP application features with the Oracle Database has recently become more visible.

Oracle recognizes the value that the tight integration between the Oracle database and the SAP application brings to our customers. Oracle's continuing commitment for both pillars is evident through the comprehensive set of database features provided and for the special HANA optimizations currently supported such as Core Data Services and Oracle Optimized Flat Cubes.

Oracle Database Version: Support Status and Roadmap

Starting from 2018, new releases of the Oracle Database software are provided annually. In addition, a new numbering schema has been implemented: Instead of the traditional version numbers, the release year is now used to designate a software version (18c, 19c, etc.). These annual software releases will be made available to SAP on Oracle customers as well.

An overview of the versions that are currently available can be found in the Oracle Database Roadmap infographic on page 5 and in figure 1 on page 7. – For additional details see SAP Notes 1174136 and 2606828.

Oracle Database 19*c*

Oracle Database 19c, certified for SAP since December 2019, is the most current long-term support release, and it is recommended for all SAP on Oracle customers. Primary Support will end on April 30, 2024; Extended Support on April 30, 2027.

Oracle Database 18*c*

Oracle Database 18c has been certified for SAP in March 2019. Primary Support will end on June 08, 2021. No Extended Support is planned.

Oracle Database 12*c* (12.2)

Primary Support for Oracle Database 12.2 (12.2.0.1) will end on November 30, 2020. Limited Error Correction is available from December 01, 2020 until March 31, 2022. – For more information see SAP Note 2855812.

Oracle Database 12*c* (12.1)

Primary Support for Oracle Database 12.1 (12.1.0.2) ended on July 31, 2018; Extended Support with Waived Fee ended on July 31, 2019. Beginning August 01, 2019, an Extended Support service contract is required. Paid Extended Support will end on July 31, 2022. – For more information see SAP Note 2428722

Oracle Database 11*g*

Primary Support for Oracle Database 11g (11.2.0.4) ended on January 31, 2015, and Extended Support with Waived Fee ended on December 31, 2018. Paid Extended Support is available until December 31, 2020[1].

Oracle Database Versions: New Features Overview

The term "base certification" has been coined during the certification of Oracle Database 12.1 for SAP. This process was split into several phases, the first of them being a certification of the new database version without any new option (base certification). In this article the term is used for different Oracle Database versions. The "base certification" section contains a discussion of basic, but important features which are not mentioned in the following sections.

Oracle Database 19c

Oracle Database 19c comes with new features that improve the Oracle/SAP integration and operating system support:

- **SQL Macros:** SQL Macros allow developers to factor out common SQL expressions and statements into reusable, parameterized constructs that can be referenced in SQL statements. Unlike PL/SQL functions, SQL Macros are evaluated at parse time, which means that at execution time context switches between SQL and PL/SQL can be avoided and SQL runtime can be reduced considerably. In SAP environments, SQL Macros are essential for application development based on CDS views. – See SAP Note 2816467.
- **Operating system support:** Oracle Database 19c is the minimum required release for the following operating system versions:
  - Oracle Linux 8
  - Red Hat Enterprise Linux (RHEL) 8
  - Microsoft Windows Server 2019.

Oracle Database 18*c*

Oracle Database 18*c* comes with enhancements and new features that improve performance, reliability and ease of management. Among the features certified for SAP environments are:

- **In-Memory Dynamic Scans** automatically and transparently parallelize table scans by using lightweight process threads. IM dynamic scans automatically use idle CPU resources to scan IMCUs in parallel and maximize CPU usage. When CPU resources are available, applications such as SAP BW can get even faster analytic query results automatically. IM dynamic scans are more flexible than traditional Oracle

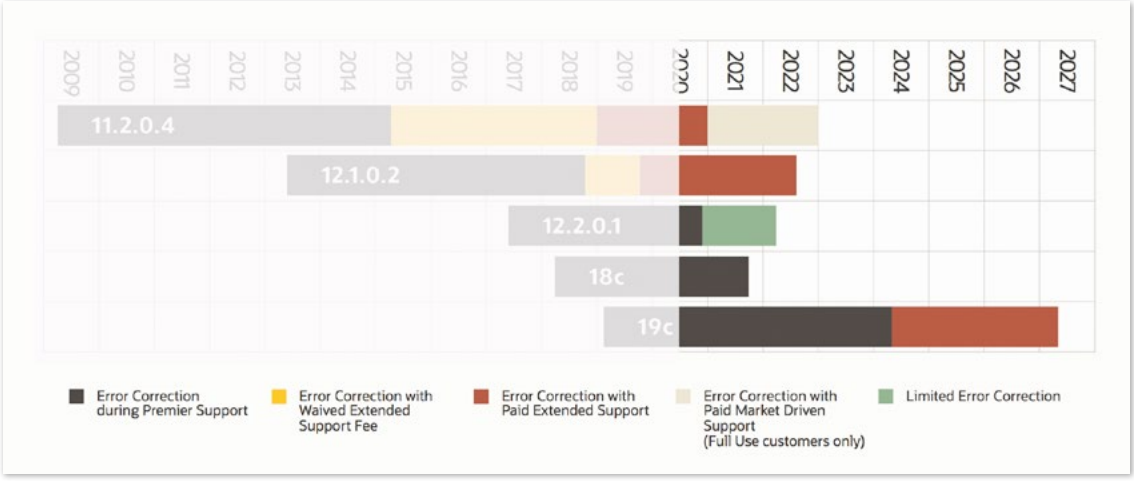


Figure 1: Oracle Database version support

[1]All statements in this section are as of June 2020. Please keep in mind that these dates are subject to change at any time.

parallel execution, although the two are not mutually exclusive. Dynamic scans use multiple lightweight threads of execution within a process.

- **In-Memory Optimized Arithmetic:** The Oracle Database `NUMBER` data type has high fidelity and precision. However, `NUMBER` can incur a significant performance overhead for queries because arithmetic operations cannot be performed natively in hardware. The In-Memory optimized number format enables native calculations in hardware for segments compressed with the QUERY LOW compression option.
- A **Polymorphic Table Function (PTF)** is a new type of table function whose return type is determined by the arguments passed into the PTF. Useful when SQL developers and database administrators want to provide generic extensions which work for arbitrary input tables or queries, it perfectly matches the `ABAP SELECT FOR ALL ENTRIES` clause.

Oracle Database 12c Release 2

Oracle Database 12.2 provides numerous enhancements of existing features or options, in particular features and options released in Oracle Database 12.1 for the first time. Examples are:

- **In-Memory Fast Start:** Ordinarily, when an instance is restarted, the in-memory column store must be rebuilt from scratch, a process referred to as in-memory populate. This process can be CPU-intensive, since it must convert row-format data into compressed columnar data. With Oracle Database 12.2, the In-Memory Fast Start mechanism can significantly reduce the total time required for population by keeping a checkpointed copy of the column store on disk. As a result, when the instance is restarted, the checkpointed copy can be directly read back into memory without requiring any transformation of the data.
- **Online Tablespace Encryption:** In older releases, only new tablespaces could be encrypted. Existing data had to be exported and re-imported. Oracle Database 12*c* Release 2 allows encryption of existing tablespaces while they are online and in read-write mode. Starting with Oracle Database 12.2, it is also supported to encrypt Oracle-supplied tablespaces (SYSTEM, SYSAUX, etc.) in addition to the tablespaces containing user/application data.
- **Operating System Support:** Oracle Database 12*c* Release 2 (12.2) is the minimum required release for the following operating system versions:
  - Microsoft Windows Server 2016
  - SUSE Linux Enterprise Server (SLES) 15.

Oracle Database 12c Release 1

Oracle Database 12c (12.1) came with two major new options and important enhancements to an existing option, which will be discussed later in this article and in other articles: Oracle Database In-Memory (see page 10), Oracle Multitenant (see page 15), and support for Information Lifecycle Management provided by Oracle Advanced Compression (see page 14). In addition, the following new features should be noted:

- **Advanced Index Compression** is a new form of index compression which is more efficient than standard Index Key Compression and requires less administration effort. Advanced Index Compression works well on all supported indexes, including those that are not good candidates with Index Key Compression.
- **Advanced Network Compression** can be used to compress the data to be transmitted at the sending side and then uncompress it at the receiving side to reduce the network traffic. Advanced Network Compression allows transmission of large data in less time. It improves SQL query response time and saves bandwidth (see SAP Note 2138262).
- Data Guard – the functionality needed to set up stand by databases – is included in Oracle Database Enterprise Edition. Active Data Guard is an add-on option. Oracle Database 11g came with additional features such as Automatic Block Repair and Fast Incremental Backup. **Active Data Guard Far Sync**, the main new feature with Oracle Database 12*c*, allows customers to combine high performance (a characteristic of asynchronous log shipping) and zero data loss (a characteristic of synchronous log shipping) across large distance WANs. For details see the article "Implementing a Data Management Infrastructure for SAP with Oracle Database Options and Packs" ("Data Guard and Active Data Guard" section) on page 25-26.
- Oracle Recovery Manager (RMAN) provides a comprehen sive foundation for efficiently backing up and recovering the Oracle Database. **Cross Platform Backup and Restore** allows you to transport data across platforms by using full and incremental backup sets. To perform cross-platform backups using backup sets, the destination database must be Oracle 12c or later. This newly added feature simplifies platform migration and minimizes read-only downtime on the source database.
- While RMAN remains the most popular tool to perform Oracle Database backups, another commonly used method for taking database backups is to create a storage snapshot of all files in the database. **Storage Snapshot Optimization** enables you to use third-party technologies to take a storage snapshot of your database without the need for putting the database in BACKUP mode.

- On Microsoft Windows, Oracle Database 12c supports the use of an Oracle Home User, which can be specified at installation time. The Oracle Home User is introduced to host Oracle Services for greater security using a low privileged non-administrator account. For more information see SAP Note 1915302.

Base Certification and Application Optimization

Theoretically speaking, implementation of Oracle support for SAP application optimizations is an ongoing project that runs completely independent from the certification of Oracle Database versions. However, in some cases certain version-specific features may be or are required.

A particularly interesting example is discussed in SAP Notes 1835008 and 1892354: Several application optimizations implemented by SAP can only be used, if some tables traditionally implemented as cluster tables are declustered. As the data in these cluster tables is normally stored in a compressed manner by SAP, customers find that the tables can grow considerably when they are converted to transparent tables.

Unfortunately some of the declustered tables have more than 255 columns. In those cases Oracle Database 11*g* Advanced Compression could not be used to reduce their size, because in this version structured table data compression (OLTP compression) was not supported for tables with more than 255 columns.

In Oracle Database 12*c* Advanced Compression, the 255-columns limit is removed, and the table compression without this limit does not exist anymore, and the enhanced compression feature has been made available for SAP customers immediately with the base certification. Therefore, with the Oracle Database 12c (and higher) Advanced Compression, it is possible to compress and manage the data residing in these very wide tables.

SAP Application Optimization: Pushdown of Data-intensive Computations from Application Layer to Data Layer

Many people believe that SAP's decision to abandon the least common denominator strategy and to optimize their applications for HANA in mind are seen as a threat by Oracle. And it is certainly true that in the SAP world HANA is a competitor of the Oracle Database. However, in many cases SAP's new application optimizations are greeted with a sigh of relief by Oracle employees as well as by Oracle customers. Taking SAP Core Data Services (CDS) as an example, it is easy to explain why.

The main questions behind Core Data Services are:
What is a database? What can it do? And what can it not do?

The traditional answer to these questions claims that a database is nothing but a dumb data store. It is a container that can permanently store data, but that's it. Whenever a customer wants to do something useful with the data, it must be transferred to the application server, because the intelligence sits in the application server.

Traditional SAP applications are based on this very concept. The disadvantages are obvious: If the sum of 1 million values needs to be calculated and if those values represent money in different currencies, 1 million individual values are transferred from the database server to the application server – only to be thrown away after the calculation has been done. The network traffic caused by this approach is suboptimal and suffers with poor performance.

More than 25 years ago, the developers of the Oracle Database asked: Wouldn't it be nice, if this sum could be calculated on the database server side? Would this not improve the answer to the question what a database is: A database is not only a data store, it can also store and execute procedures working with the data – pieces of code that originally were part of the application running on the application server, but are now moved to the database server. So the application is split into two tiers, one of them running on the application server, the other one on the database server, and therefore the database server is an application tier.

The Oracle developers did not only ask questions or come up with a new concept. They also built a new database version that was able to store and execute database procedures (Oracle 7, released in 1992).

However, at that time the Oracle Database was the only database that could process application logic at the database layer. Stored procedures were not part of the least-common-denominator feature subset, and therefore SAP declined to use them.

20 years later, SAP started to promote HANA, One of the first things they discovered was that their own applications were the worst enemies of the new in-memory database architecture. If an application believes that a database is essentially a dumb data store, that only itself can do calculations efficiently and therefore individual values need to be transferred over the network, actively destroys all potential benefits of an in-memory database. At that time, SAP realized that they had to abandon the least common denominator strategy and its counterpart, the dumb data store concept.

As a response to this insight, SAP developed the "Push down" strategy: push down code that requires data-intensive computations from the application layer to the database layer. They developed a completely new programming model that allows ABAP code to (implicitly or explicitly) call procedures stored in the database. And in order to prevent pure chaos, they defined a library of standard procedures. This library is called Core Data Services (CDS). And they agreed to make this library available for non-HANA databases, too, if those databases support stored procedures.

The 20 years between the release of Oracle 7 and the release of SAP Core Data Services explain the sighs of relief breathed by Oracle customers and employees: The performance gains achieved by SAP's push-down strategy would have been possible 20 years earlier. Better late than never.

A second example for the same strategy is **FEMS Pushdown**. FEMS queries can be thought of as a spreadsheet and query conditions that define how to calculate the cell values. FEMS Pushdown, which allows all calculations to be done in the database, can reduce database time, network traffic, and application server time considerably. It is supported for the Oracle Database as of July 2019. For more information see SAP Note 2816467.

Oracle Database Option: Oracle Database In-Memory

Oracle Database 12c (and higher) comes with a Database In-Memory option, however it is not an in-memory database. Supporters of the in-memory database approach believe that a database should not be stored on disk, but (completely) in memory, and that all data should be stored in columnar format. It is easy to see that for several reasons (among them data persistency and data manipulation via OLTP applications) a pure in-memory database in this sense is not possible. Therefore, components and features not compatible with the original concept have silently been added to in-memory databases such as HANA. Oracle has chosen the opposite strategy: Data can be populated into an In-Memory Column Store whenever this makes sense. In all other cases, data are stored and handled as it always has been.

For more information on the concepts of Oracle Database In-Memory see the article "Implementing a Data Management Infrastructure for SAP with Oracle database Options and Packs", in particular the sections „Oracle Database In-Memory", page 22 and „Summary", page 34.

Oracle Database In-Memory was certified for SAP in June 2015. Unlike similar options offered by competitors, the use of Oracle Database In-Memory is not limited to SAP Business Warehouse (SAP BW). It is supported for all SAP applications based on SAP NetWeaver, including typical OLTP applications. However, this does not mean that it is always a good idea to use Oracle Database In-Memory. This option is a solution for a specific problem – or for a certain class of problems. It cannot solve all problems. It cannot improve performance in all cases. If used in an inappropriate manner, it can even – like a pure in-memory database – degrade system performance. Therefore, the SAP applications that can benefit from data being loaded into the column store must be selected carefully.

Applications must be selected, individual tables must be selected – the implementation of Oracle Database In-Memory in SAP environments seems to be difficult. However, early adopters consistently mention as their very first experience that Oracle Database In-Memory for SAP can be implemented quickly and easily. This seems to be counterintuitive, but it is not.

First, many customers are already aware of the queries and jobs that take too much time to complete, and they know which tables are involved. In those cases the task to select appropriate SAP applications and tables is trivial.

Second, for customers who do not want to implement Oracle Database In-Memory in order to fix specific issues, but prefer a general approach, Oracle provides an In-Memory Advisor – a wizard that analyzes the workload of a particular system and recommends tables to be populated into the column store based on the amount of memory that is available. (This means that the frequently asked question „How much memory do I need in order to use Oracle Database In-Memory?" is completely meaningless. It's the other way round: You tell Oracle how much memory you have, and the advisor will let you know how that amount of memory can be used in the most efficient way.)

Third, once the relevant tables are determined, everything is easy and breathtakingly fast: By issuing an `ALTER TABLE <table_name> INMEMORY` statement you declare that those table data should be available in the column store and from this point on everything else happens automatically in the background.

Finally, unlike the migration to an in-memory database such as HANA, the implementation of Oracle Database In-Memory does not require a revolution: no new hardware, no new operating systems, no new database. Customers can continue to use the existing infrastructure, and what administrators need to know about Oracle Database In-Memory can be learned within a few hours.

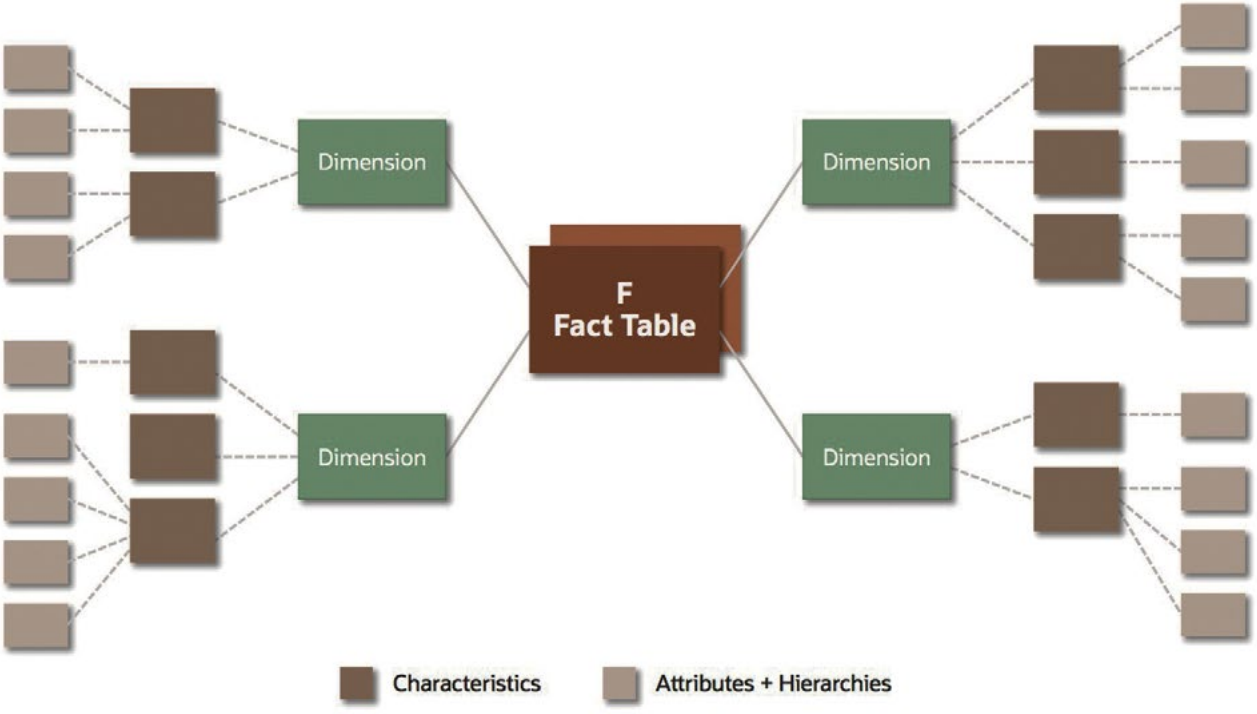


Figure 2: Traditional "star" (= extended snowflake) schema

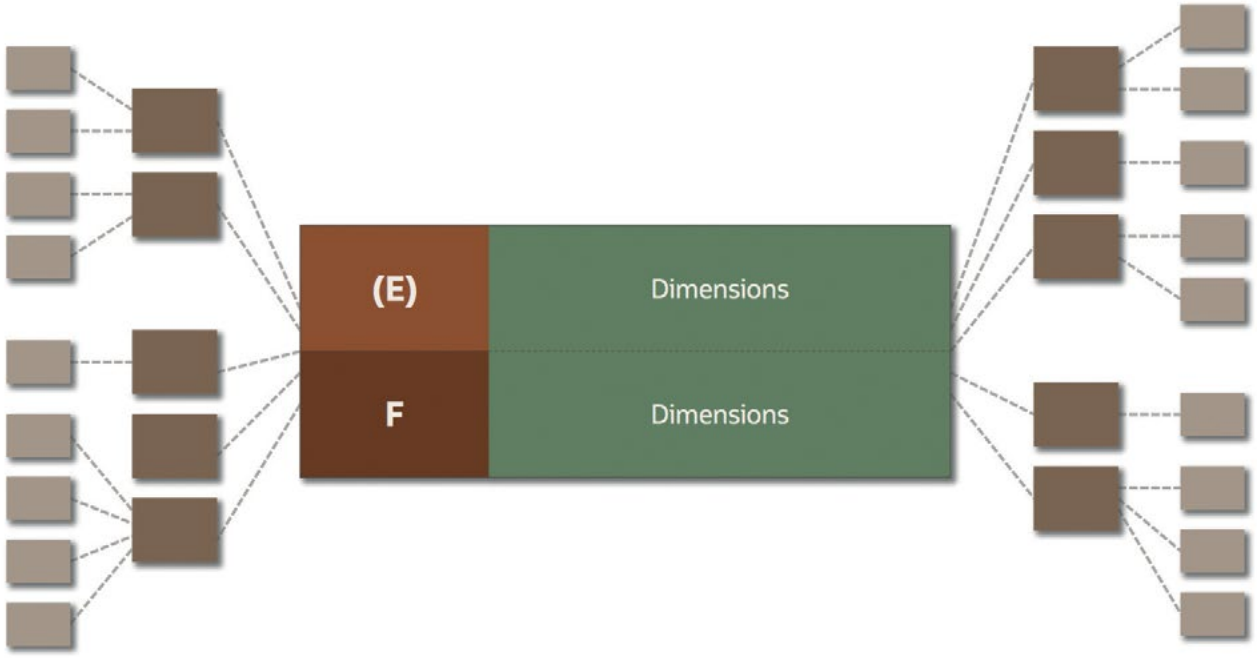


Figure 3: New flat cube design

SAP Application Optimization: Flat Cubes

When the certification of Oracle Database In-Memory for SAP was announced in June 2015, the announcement included a couple of restrictions. In particular, it was strongly recommended not to drop any standard indexes or aggregates. This caused some disappointment, because from a pure Oracle perspective indexes are not needed anymore when the base tables are populated into the column store, and can therefore be dropped.

But in this case (as in all other cases described in this article) the Oracle/SAP development team, which is responsible for the integration of SAP and Oracle technologies, had to follow SAP's learning curve. The situation immediately after the certification of Oracle Database In-Memory for SAP (in this case: for SAP BW) simply mirrors the early stages of SAP's project to provide SAP BW on HANA.

The disappointment mirrors SAP's experience that the traditional SAP BW data model is not compatible with the new concept of an in-memory database. Flat Cubes, which will be explained in this section, utillzed the new data model that SAP designed for HANA.

In many cases, data to be loaded into the Business Warehouse arrive as very wide records. E. g. company name, zip code, city, and street address are combined with carrier details, order number, order date, invoice number and dozens, if not hundreds of other data items in one single record. But in the early days of data warehousing, when databases were disk-based only and disk space was expensive, it was not acceptable to waste disk space for redundant data such as the company or the carrier details which occur 1000 times, if that particular company sends 1000 items, and 100,000 times, if that particular carrier is engaged to fulfill 100,000 shipments. Therefore database architects came up with a design called star schema: subsets of data which belong together (all customer details, all carrier details) are moved to separate tables, which are called dimension tables. The remaining data plus IDs pointing to the relevant entries in the dimension tables is stored in the fact table.

Such a split was not enough in all cases. E. g. a certain combination of zip code, city name and street may occur several times in the CUSTOMERS as well as in the CARRIERS table. If the same split operation is applied again, additional tables are created which, however, are not connected to the fact table, but to the dimension tables. This results in a more complex, but also (from a disk-space point of view) more efficient design, which is called snowflake schema. High-end data warehouses such as SAP BW add yet another level of detail tables, thus relying on the extended snowflake schema.

This complex architecture has been designed in order to optimize the data model for the requirements of traditional, disk-only relational databases. However the new databases with their focus on memory – and in this respect there is no difference between SAP HANA and Oracle Database In-Memory – have very different requirements.

Therefore, SAP designed a new data model for SAP BW on HANA and consequently called it HANA-Optimized InfoCubes. The simplest, but somewhat surprising description of HANA-Optimized InfoCubes is this: If the process of optimizing the SAP BW data model for disk-oriented databases led from flat and therefore wide records to the extended star schema, the process of optimizing the data model for memory-oriented databases is simply the way back from extended star to flat and wide.

Back but not all the way. HANA-Optimized InfoCubes combinethe fact table (actually: the E and F fact tables) and the dimension tables (first level of details) in one single table, whereas the small level 2 and 3 tables (characteristics, attributes and hierarchies) remain in place. This change is sufficient to improve performance and manageability considerably.

This new data model removes the main disadvantages of the previous data model without sacrificing its benefits. It is no longer necessary to split the incoming, wide records in order to distribute them over many tables – this speeds up data load. The traditional indexes are not needed anymore – this speeds up data load as well. It is no longer necessary to join the tables later on – this speeds up query processing. The main disadvantages of the flat data model that originally motivated the development of the extended snowflake schema have been the disk and memory requirements of storing redundant data. This is no longer a concern thanks to Oracle's Advanced Compression features available today that optimize the storage for data on disk as well as data in memory.

If this new data model is made available for a non-HANA database, "HANA-Optimized InfoCubes" is obviously not an appropriate name. "SAP BW Flat InfoCubes for Oracle" or simply "SAP BW Flat Cubes for Oracle" is exactly the same data model, called by a different name. It requires Oracle Database 12c or higher and Oracle Database In-Memory, as Flat Cubes outside of the Column Store do not make any sense.

Flat Cubes for SAP BW on the Oracle Database is generally available since June 2016. For more information see SAP Note 2335159.

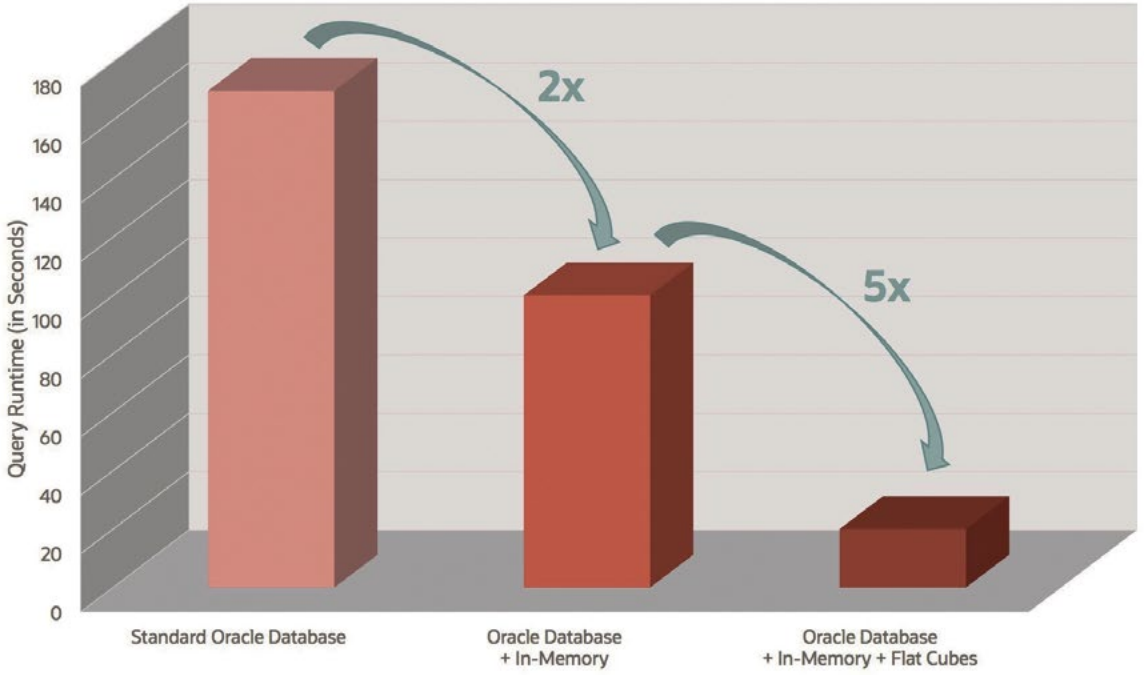


Figure 4: Performance gains with Oracle Database In-Memory and Flat Cubes for SAP BW

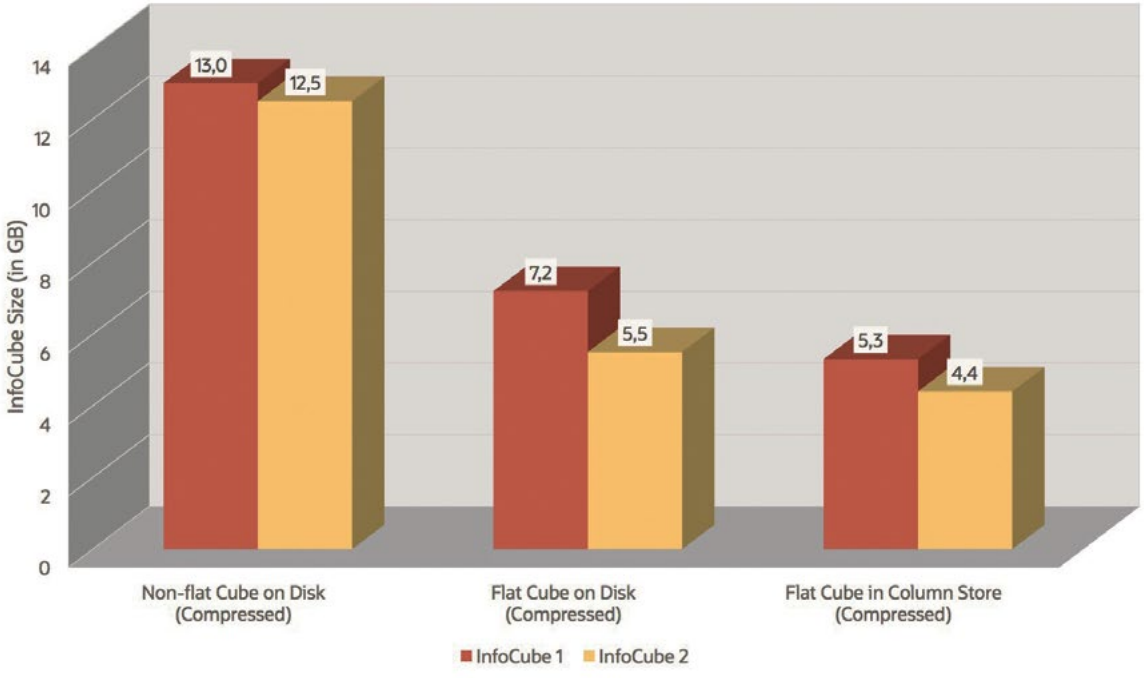


Figure 5: Disk space and memory consumption without and with Flat Cubes for SAP BW

Oracle Database Features: Deferred Compression and Information Lifecycle Management

Some of the new features in Oracle Database 12*c* Advanced Compression have already been discussed in the "Base Certification Features" and "Base Certification and Application Optimization" sections. However, two major new features are still missing, because they were not included in the base certification, but certified a few months later for SAP environments: Heat Map and Automatic Data Optimization (ADO). The basic concepts behind these two features are discussed in the article "Implementing a Data Management Infrastructure for SAP with Oracle Database Options and Packs" (see in particular the section "Advanced Compression (Oracle Database 12*c* and higher)", page 21). Therefore, we will briefly look at the SAP-specific implementation details.

Oracle Database 12*c* Advanced Compression allows customers to distinguish between current ("hot") and historical ("cold") data. However, it is not clear what exactly the words "hot" and "cold" mean. So this needs to be defined:

```
ALTER TABLE <table_name> ILM ADD POLICY
<action>
AFTER <n> DAYS OF NO MODIFICATION;
```

The third line of this SQL statement answers the question. New data is considered "hot". If it turns out that they have not been modified for a certain number of days (30, 60, 90 days), they are considered "cold" – assuming that the customer does not want to define intermediate levels such as "warm". But if we look closer, we find that the only question that has been answered so far is: When do we call data "cold"? What we still do not know (and what the database system still does not know) is: If data have cooled down – then what? What should happen? This is to be defined in line 2:

```
ALTER TABLE <table_name> ILM ADD POLICY
ROW STORE COMPRESS ADVANCED ROW
AFTER 40 DAYS OF NO MODIFICATION;
```

In this example we assume that (in this particular table) hot data is not compressed at all, and we tell the system that (a) any data not modified for 40 days should be considered cold and that (b) cold data should be compressed using the table compression algorithm provided by Oracle Database Advanced Compression.

How do we, and how does the system know that data have not been modified for 40 days? It is the job of Heat Map to provide this kind of information. Heat Map automatically tracks modification and query timestamps at the row and segment levels, providing detailed insight into how data is being accessed. *Automatic Data Optimization (ADO)*, then, automatically moves and compresses data according to user-defined policies (such as that which we have used here as an example) based on the information collected by Heat Map.

So far the ALTER TABLE statement has been used to define the ILM policy. In SAP systems where we have to deal with tens of thousands of tables, this approach would be very cumbersome. Therefore the BR*Tools (BRSPACE) use a different option provided by the Oracle Database:

```
ALTER TABLESPACE TSX DEFAULT ILM ADD POLICY
ROW STORE COMPRESS ADVANCED ROW
AFTER 40 DAYS OF NO MODIFICATION;
```

In this example we do not define a special policy for an individual table, but a default policy on the tablespace level. It is automatically applied to all tables created in this tablespace, unless a table comes with an individual policy.

Customers running Oracle Database 12*c* or higher on Oracle Engineered Systems such as Exadata can benefit from Hybrid Columnar Compression – a set of compression algorithms designed for purely historical data as an alternative to archiving. If Advanced Compression compresses data by a factor of 2 or 3, Hybrid Columnar Compression can easily achieve compression factors of 10 or 15.

In this situation, we would call data not modified for 40 days "warm", and we would reserve the word "cold" for data not changed during a considerably longer period (e.g. 6 or 12 months). We would keep the previous policy as compression tier 1 (for warm data) and add an additional policy as compression tier 2 (for cold data). And we would separate unpartitioned and partitioned tables in different tablespaces, because Hybrid Columnar Compression compresses complete partitions instead of individual blocks:

```
ALTER TABLESPACE TSY DEFAULT ILM ADD POLICY
ROW STORE COMPRESS ADVANCED ROW
AFTER 40 DAYS OF NO MODIFICATION;
ALTER TABLESPACE TSY DEFAULT ILM ADD POLICY
COLUMN STORE COMPRESS FOR QUERY LOW ROW
LEVEL LOCKING SEGMENT
AFTER 6 MONTHS OF NO MODIFICATION;
```

Oracle Database Option: Oracle Multitenant

Oracle Multitenant helps customers reduce IT costs by simplifying consolidation, provisioning, upgrades, and more. It is supported by an architecture that allows a container database (CDB) to hold and manage many pluggable databases (PDBs) (see article "Implementing a Data Management Infrastructure for SAP with Oracle Database Options and Packs", in particular section "Oracle Multitenant", page 26).

With Oracle Multitenant, multiple existing databases may be converted to PDBs and consolidated into a single CDB. A PDB is a self-contained, fully functional Oracle Database. From an application's point of view nothing has changed in any way, and that's very important because it means that no application changes are required to adopt this architecture. From an application's point of view, the PDB is the database. However, from an operational point of view the CDB is the database.

The CDB represents a single, consolidated operating environment. There is a single set of background processes and a single shared memory area (SGA), shared by all of the PDBs in the CDB. This architecture eliminates replication of overheads, making most efficient use of available resources. What this means is that you can minimize capital expenses (CapEx) because you can consolidate more applications per server. From an operational perspective, you can manage all these consolidated PDBs collectively, greatly reducing operating expenses (OpEx). This applies to things like backups, configuration of high availability, application of patches and upgrades. These CapEx and OpEx reductions are part of how Multitenant delivers on the promise of Cloud computing.

Oracle Multitenant is the architecture for the next-generation Database Cloud. Multitenant delivers true *economies of scale*. The expensive model of a VM containing a database is replaced by a pluggable database (PDB). Because there is negligible intrinsic cost to a PDB, the cost of each SAP system' s PDB is reduced to the actual work they do.

The Oracle Multitenant architecture can be used for all SAP NetWeaver-based applications with the only exception that a mix of SAP OLAP (BW) and SAP OLTP (ERP, CRM, ...) systems in the same container database is not supported. For database administrators, the following tool support is available:

- As of version 1.0 SP 19, SWPM supports the creation of container databases (CDBs) and pluggable databases (PDBs). SWPM must be used for these tasks in order to guarantee compatibility of the created databases (directory paths, file names, etc.) with BR*Tools. For details see SAP Note 2336881.
- In most cases, customers will not create new databases, but convert existing stand-alone (non-CDB) databases to pluggable databases. SAP Note 2335850 describes the supported procedure for this kind of transformation.
- As of version 7.40 patch 24, BR*Tools support Oracle Multitenant. New configuration parameters, commands, and command options allow administrators to specify the target database(s) for operations initiated via the familiar BRCONNECT, BRSPACE, BRBACKUP/BRARCHIVE or BRRESTORE/ BRRECOVER commands.

SAP Application Optimization Support

APPLICATION OPTIMIZATION TYPES

Workload Distribution Optimizations

Application Development Optimizations

Data Model Optimizations

WORKLOAD DISTRIBUTION OPTIMIZATIONS

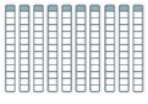
ABAP Core Data Services:
Push data-intensive computations from application layer to database layer

FEMS Pushdown:
Push complete FEMS queries („spreadsheets“) to database layer

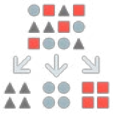
APPLICATION DEVELOPMENT OPTIMIZATIONS

CDS-based Application Development:
Build your own Fiori applications based on CDS views (Whitepaper “ABAP Core Data Services on any DB“)

DATA MODEL OPTIMIZATIONS



Flat Cubes:
Optimize SAP BW Cubes for In-Memory Computing(



Declustering/Depooling:
Convert clustered and pooled tables to transparent tables

Oracle Database Support for SAP Application Optimizations – Information Resources

Document Type	Document Number	Document Title
Workload Distribution Optimizations : Core Data Services (CDS)		
White Paper	---	„ABAP Core Data Services. SAP Business Suite – Best Practices Guide“ [https://tinyurl.com/SAP-ABAP-CDS-on-anyDB]
Documentation	---	• https://tinyurl.com/SAP-ABAP-CDS-Docu-750 • https://tinyurl.com/SAP-ABAP-CDS-Docu-751 • https://tinyurl.com/SAP-ABAP-CDS-Docu-752
Workload Distribution Optimizations : FEMS Pushdown		
SAP Note	2816467	Support for FEMS Pushdown on Oracle
Application Development Optimizations		
White Paper	---	„ABAP Core Data Services. SAP Business Suite – Best Practices Guide“ [https://tinyurl.com/SAP-ABAP-CDS-on-anyDB] SQL Macros: CDS Views with Parameters
SAP Note	2801989	SQL Macros: CDS Views with Parameter
Data Model Optimizations : Flat Cubes (BW)		
SAP Note	2335159	Flat Cubes for SAP BW on Oracle DB
SAP Note	2711358	Conversion of Semantically Partitioned Objects to a Flat Format
Data Model Optimizations : Declustering/Depooling (OLTP)		
SAP Note	1835008	Activate Database Performance Optimizations for SAP ERP
SAP Note	1892354	SAP Strategy for Cluster and Pool Tables