ORACLE DATABASE 12*c* FOR SAP: LATEST DATABASE TECHNOLOGY AND SUPPORT FOR APPLICATION OPTIMIZATIONS

Strategy and Roadmap

Integration Strategy

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.

The need to *integrate Oracle Database features with the SAP environment* has always been visible. It was particularly obvious, whenever 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.

This is by no means a matter of the past. The current plan to make Oracle Multitenant available for SAP customers is a similar architectural revolution and requires 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) have been developed on the Oracle Database. And when, later on, SAP started to support IBM DB2 and Microsoft SQL Server as well, 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 to 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. It is recognized as the need to integrate SAP application optimizations designed with HANA in mind with the Oracle Database or to support SAP application optimizations by the Oracle Database.

Today, both pillars supporting the SAP on Oracle Database strategy are clearly visible: 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.

Certification Roadmap

Standard maintenance for Oracle Database 11*g* (11.2.0.4) ended on January 31, 2015. The Extended Support phase, which is restricted to three years, has started as of February 2015. For Oracle Database version 11.2.0.4 Oracle offers Extended Support until May 31, 2017 at no additional cost. (For more information see SAP Note 2098258.)

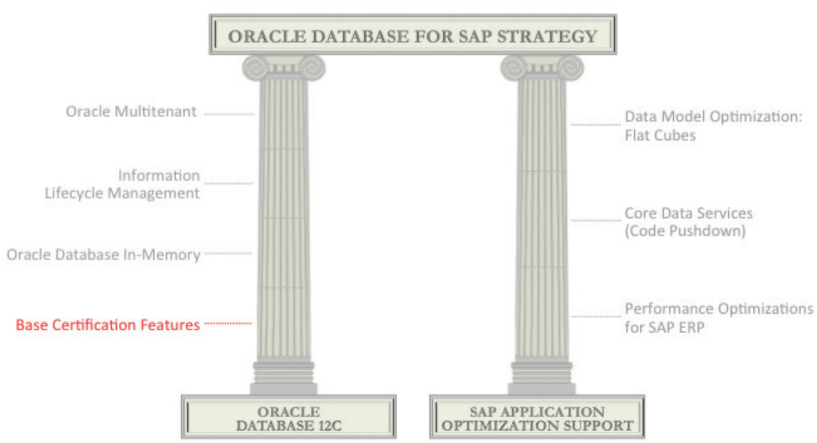
For the joint benefit of our customer base, Oracle and SAP agreed to split the certification process into several phases. Segmenting the rollout in stages, allows for the Oracle Database 12*c* to be made generally available as early as possible and ensures sufficient overlap with Oracle Database 11*g*. Oracle and SAP have broken the "terminal release" custom (certification of release x.2 only; no certification of release x.1) valid for more than 10 years.

- P*hase 1*, called *Base Certification*, has already been completed in March 2015. Included are all features and options that have been previously available in Oracle Database 11*g* plus several Oracle Database 12*c* features which are either completely transparent or require only minimal integration effort.

- *Phase 2*, completed in June 2015, is the first to offer a major new Oracle Database 12*c* option: Oracle Database In-Memory, a completely new option in Oracle Database 12*c*.
- *Phase 3*, completed in December 2015, has added the new *Information Lifecycle Management (ILM)* features included in Oracle Database 12*c* Advanced Compression, as well as *Hybrid Columnar Compression (HCC)* with row-level locking on Oracle Exadata and Oracle SuperCluster.
- *Phase 4*, Oracle Multitenant completed the certification process for SAP customers. This option allows for the consolidation of many databases into a single container database. It is based on a completely new database architecture and therefore requires a considerable Oracle/SAP integration effort.

Theoretically speaking, implementation of Oracle support for SAP application optimizations is an ongoing project that runs completely independent from the Oracle Database 12*c* certification process. However, in some cases certain Oracle Database 12*c* features may be or are required.

- *SAP Core Data Services*, originally released with SAP NetWeaver 7.40 (SP 05), is supported with Oracle Database 11*g* and Oracle Database 12*c*. No particular features or options are required. (See SAP Note 1951491)
- Many *data model optimizations* will originally result in higher disk space requirements. In those cases the use of table compression is certainly not required, but recommended. And as a considerable percentage of the tables involved contain more than 255 columns, Oracle Database 12*c* is the only version that allows customers to compress all relevant tables.
- *Flat Cubes* can only be used with Oracle Database 12*c* and Oracle Database In-Memory.



Base Certification Features

Advanced Compression

Customers running SAP applications on Oracle Database 11*g* can already choose from a variety of compression features: Index Key Compression and Compressed Index-Organized Tables (IOTs) are standard database features. OLTP Compression for structured data and SecureFiles Compression for unstructured data are provided by Oracle Database 11*g* Advanced Compression.

Oracle Database 12*c* Advanced Compression comes with many new features. Some of them are related to information lifecycle management support, which is available for SAP customers as of certification phase 2. However, there are several Advanced Compression Option features included in phase 1 that can be utilized today within SAP environments:

Advanced Index Compression is a new form of index compression. Creating or rebuilding an index using Advanced Index Compression reduces the size of unique and non-unique indexes, while still providing efficient access to the indexes. Benefits include:

- Advanced Index Compression works well on all supported indexes, including those indexes that are not good candidates with the existing Index Key Compression.
- Advanced Index Compression works at the block level to provide the best compression for each block. This means that users don't need knowledge of data characteristics – Advanced Index Compression automatically chooses the right compression per block.

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 reduces the size of the data to be transmitted over a network connection. Benefits include:

- Increased effective network throughput: Compression allows transmission of large data in less time. SQL query response becomes faster due to the reduced transmission time.
- Reduced bandwidth utilization: Compression saves bandwidth by reducing the data to be transmitted, allowing other applications to use the freed-up bandwidth. This also helps in reducing the cost of providing network bandwidth.

Advanced Network Compression can only be used if both the Database/Instant Client and the Database Server are upgraded to 12.1.0.2.

For more information about Oracle Database 12*c* Advanced Compression for SAP Systems see SAP Note 2138262.

Active Data Guard

Data Guard – the functionality needed to set up standby databases – is included in Oracle Database Enterprise Edition. Active Data Guard is an extra option. In Oracle Database 11*g* it offers 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 data shipping) and zero data loss (a characteristic of synchronous data 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.

Backup and Recovery

Oracle Recovery Manager (RMAN) provides a comprehensive foundation for efficiently backing up and recovering the Oracle Database. It is designed to work intimately with the server, providing block-level corruption detection during backup and restore operations. RMAN optimizes performance and space consumption during backup through the use of file multiplexing and backup set compression. RMAN also integrates with Oracle Secure Backup, as well as third party media management products, for tape backup. **Cross Platform Backup and Restore** enables you to transport data across platforms by using full and incremental backup sets.

The Oracle Database 12*c* allows you to transport data across platforms using either full or incremental backups, using image copies or backup. To perform cross-platform backups using backup sets, the destination database must be Oracle 12*c* 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. Mount the snapshot on a different server (other than the one that runs the production database) and copy the data to a tertiary storage such as tape, thus offloading the backup processing from the production server. **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.

Snapshots taken this way are "crash-consistent", provided the storage product adheres to specific guidelines outlined in Oracle documentation. Crash-consistent backups can be opened and used after undergoing a *full crash recovery*. However, they cannot be reliably used for *point-in-time recovery*, as the redo logs do not contain sufficient information to remove the data files' inconsistencies. Alternatively, snapshots taken in backup mode, i.e.

`ALTER DATABASE [BEGIN|END] BACKUP`,

remove the point-in-time recovery restriction. However, each database needs to be placed in this mode before the snapshot is taken, and taken out of this mode when the snapshot completes. This complexity is magnified when the procedure must be done for tens, hundreds, or thousands of databases. In addition, during this mode, whole block images are written to redo logs as they are changed, inducing additional I/O activity.

With Oracle Database 12*c*, the

`RECOVER ... SNAPSHOT TIME`

command, storage snapshots taken without the database in backup mode can be recovered in one step, whether to the current time or a specific point-in-time after the snapshot was taken, without any additional procedures. By supporting all types of recovery operations using these snapshots, this optimization effectively eliminates the need for backup mode and its associated complexity and overhead, freeing the DBA's time to focus on more critical production tasks.

ACFS Support

If you work with Exadata, you are probably aware that ACFS has not been supported until now. **ACFS is now supported on Exadata**, if you are running Grid Infrastructure version 12.1.0.2 or later. In SAP environments this can be used for SAP shared file systems (`/sapmnt`, etc.).

However, it is not meant as an alternative for running databases on ASM. Oracle Databases have to stay on ASM using the Exadata Storage nodes. This is the only supported configuration.

High Availability Network File Storage (HANFS) for Oracle Grid Infrastructure provides uninterrupted service of NFS V2/V3 exported paths by exposing NFS exports on Highly Available Virtual IPs (HAVIP) and using Oracle Clusterware agents to ensure that the HAVIPs and NFS exports are always online. If a cluster node fails, the HAVIP and NFS exports are automatically migrated to a surviving node.

The HANFS feature enables highly available NFS servers to be configured using Oracle ACFS clusters. The HANFS cluster configurations may be built from your existing infrastructure or commodity servers and storage. This provides network services similar to 'network filer' at a fraction of the cost.

Performance

The UNION and UNION ALL operators in SQL statements connect two or more branches (e.g. subqueries): <branch1> UNION <branch2>. Traditionally, in queries of this type, the branches execute one after another, meaning that only one branch is executed at a certain point in time, followed by the next branch, and so on in a serial manner. Oracle Database 12*c* introduces **Concurrent Execution of Union and Union All Branches**, meaning that one set of parallel servers will be executing one branch, a second set of parallel servers will be executing a different branch, and so on, all at the same time.

The ability to parallelize these branches will lead to much faster statement execution. In particular, it will improve SAP BW performance.

Manageability and Availability

Database administrators can now perform a variety of additional reorganization operations online using BR*Tools:

Online Move Partition: Starting with Oracle Database 12*c*, the

`ALTER TABLE ... MOVE PARTITION`

operation functions as a non-blocking online DDL command, while DML operations continue to execute uninterrupted on the partition that is being moved. Additionally, global indexes are maintained when a partition is moved, so that a manual index rebuild is no longer required.

Move Datafile Online: Prior to Oracle Database 12*c*, moving datafiles has always been an offline task. There were certain techniques you could employ to minimize that downtime, but you couldn't prevent it completely. Oracle Database 12*c* includes an enhancement to the `ALTER DATABASE` command to allow datafiles to be moved online.

Rebuild Index-Organized Tables: Because index-organized tables are stored as B-tree indexes, you can encounter fragmentation as a consequence of inserts, updates and deletes. However, you can use the

```
ALTER TABLE ... MOVE ONLINE
```

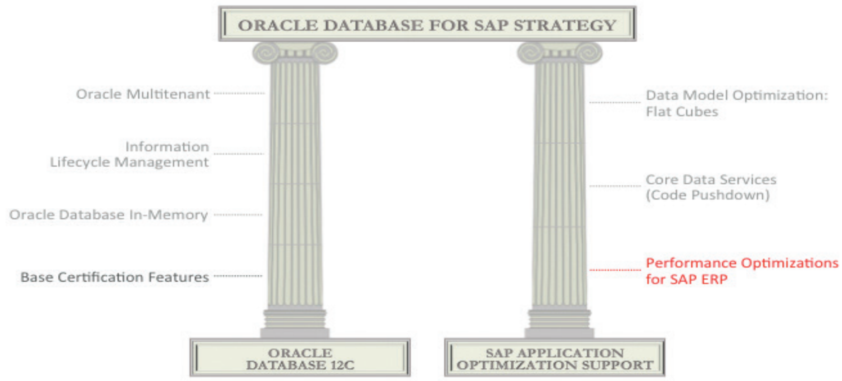
statement to rebuild an index-organized table and reduce this fragmentation.

For more information see SAP Notes 1856270 and 2087004.

Microsoft Windows

Oracle Database 12*c* 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. Oracle Home User can be a Windows Built-in Account or a standard Windows User Account (not an Administrator account). This account is used for running Windows services for Oracle Home. For enhanced security, Oracle recommends using a standard Windows User Account (instead of a Windows Built-in Account).

For more information see SAP Note 1915302.

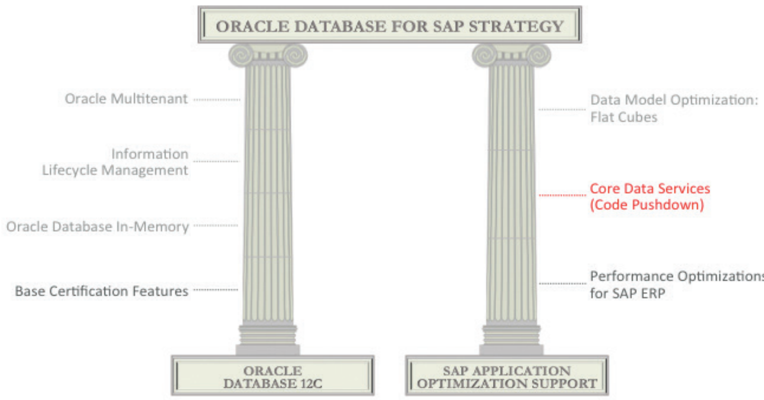


Base Certification and Application Optimization

In Oracle Database 11*g*, structured table data compression (OLTP compression) is 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 is available for SAP customers since the base certification.

On the surface this seems to be a minor improvement, however tables having more than 255 columns are commonly found in almost all SAP systems.

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. The Oracle Database 11*g* Advanced Compression could not help to reduce their size. With the Oracle Database 12*c* Advanced Compression Option, it is now possible to compress and manage the data residing in these very wide tables.



Core Data Services

Many people believe that SAP's decision to abandon the least common denominator strategy and to optimize their applications with 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 question behind Core Data Services is: What is a database? What can it do? And what can it not do?

The traditional answer to this question 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 responsible for the bad performance.

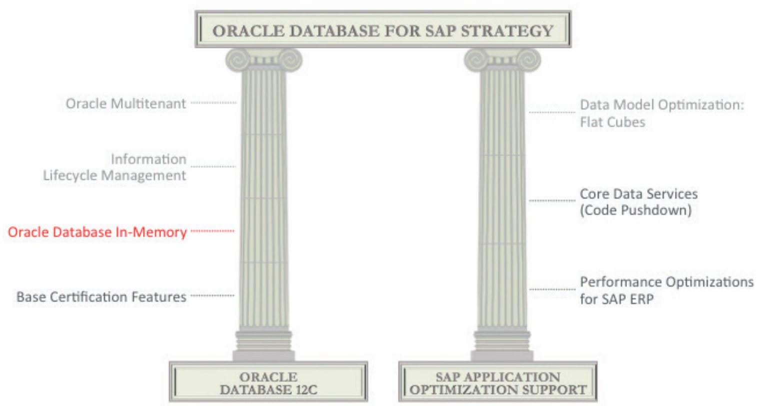
More than 25 years ago the developers of the Oracle Database asked: Would not it be nice, if this sum could be calculated on the database server side? Would not this improve the performance dramatically? They came up with a different answer to the question what a database is: A database is not only a data store. Not only can it store data, it can also store and execute procedures working with those 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*.

But the Oracle developers did not only ask questions. They did not only 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 1992). However, at that time the Oracle Database was the only database that could be used as an application tier. Stored procedures were not part of the least-common-denominator feature subset, and therefore SAP declined to use them.

When, 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. As long as an application believes that a database is essentially a dumb data store, that only itself can do calculations efficiently and that therefore individual values need to be transferred over the network, it 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. But better late than never.



Oracle Database In-Memory

Oracle Database 12*c* 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 in the past. (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 has been 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 set 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, beta customers and 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 can be explained easily.

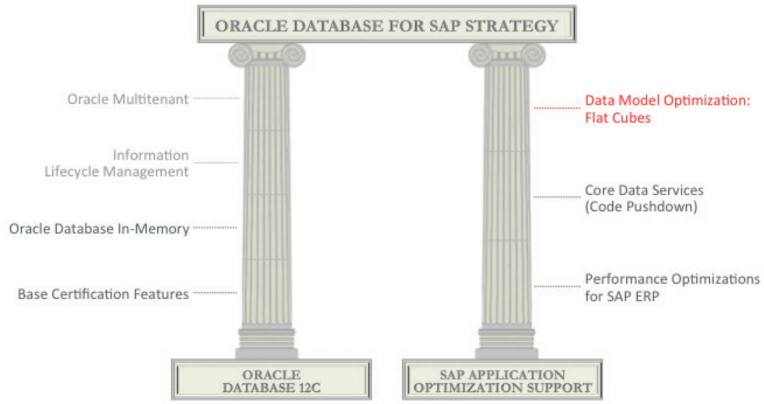
First, many customers are already aware of the queries and jobs that take too much time to complete, and they know which tables are involved. In those cases the task to select appropriate SAP applications and tables is trivial.

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 XXX 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.



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.

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

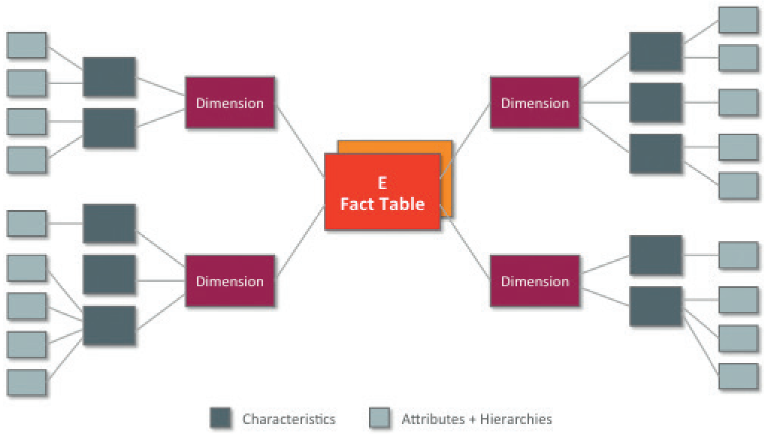
In many cases, data to be loaded into the Business Warehouse arrive as very *wide records*. E. g. company name, zip code, city, and street address are combined with carrier details, order number, order date, invoice number and dozens, if not hundreds of other data items in one single record. But in the 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 orders, 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*.

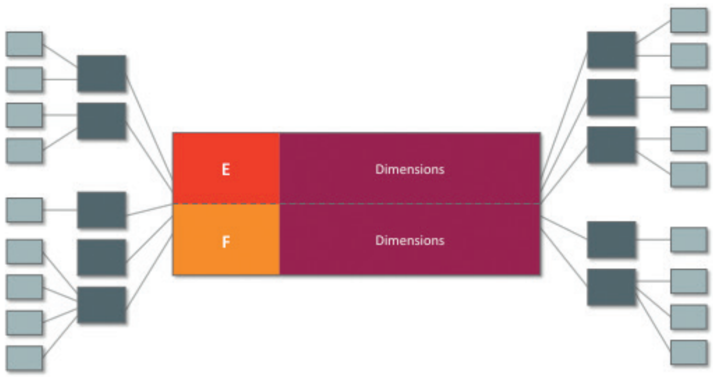
This complex architecture has been designed in order to optimize the data model for the requirements of traditional, disk-only relational databases. However the new databases with their focus on memory – and in this respect there is no difference between SAP HANA and Oracle Database In-Memory – have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

Not all the way back, however. HANA-Optimized InfoCubes combine the 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.



Traditional "star" (= extended snowflake) schema

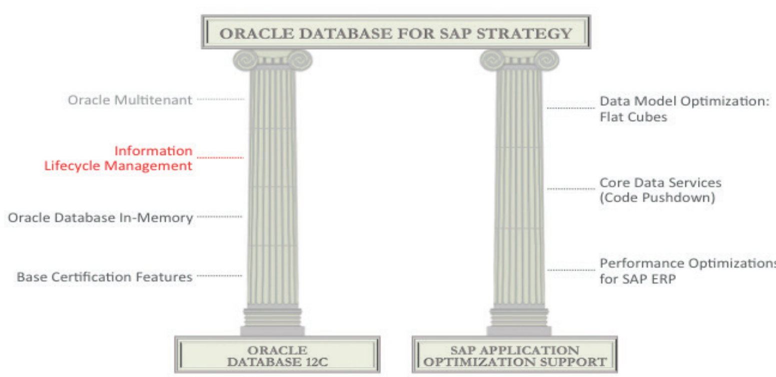


New flat cube design

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. And the main disadvantage of the flat data model, which originally motivated the development of the extended snowflake schema, the disk and memory requirements of redundant data, is no longer an issue thanks to the compression features that are available today for data on disk as well as for 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 12*c* and Oracle Database In-Memory, as Flat Cubes outside of the Column Store do not make any sense.

Flat Cubes for the Oracle Database is available since February 2016 for pilot customers. The plan is to make it generally available by June 2016.



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 (December 2015) 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*)", page 20. Here, 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. Initially, 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 are 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 are 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 12*c* 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, however, where we have to deal with tens of thousands of tables, this approach would be very cumbersome. Therefore SWPM (SAPinst) uses 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;
```

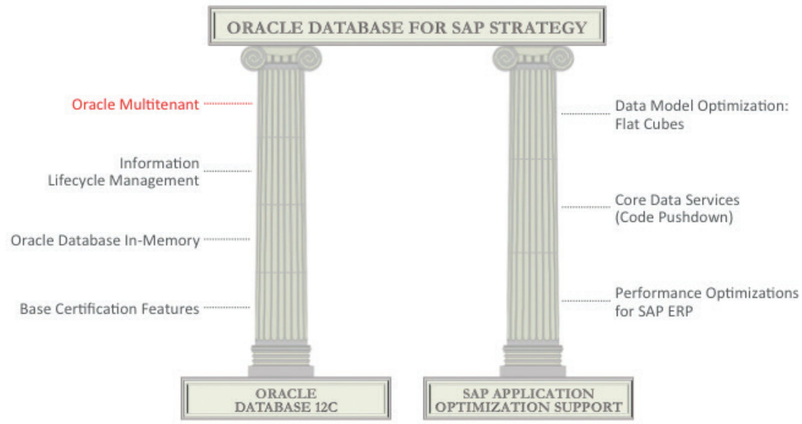
In this example we do not define a special policy for an individual table, but a *default policy* on the *tablespace* level. It is automatically applied to all tables created in this tablespace, unless a table comes with an individual policy.

Customers running Oracle Database 12*c* on an Oracle Engineered System (Exadata, SuperCluster) 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 MODIFI-
CATION;
```



Oracle Multitenant

For information on Oracle Multitenant see the article "Implementing a Data Management Infrastructure for SAP with Oracle Database Options and Packs" (in particular the section "Oracle Multitenant", page 26).

SAP Notes related to Oracle Database 12*c*

DB: Features: Overview
105047, Support for Oracle Functions in the SAP Environment
1914631, Central Technical Note for Oracle Database 12*c* Release 1 (12.1)
2133079, Oracle Database 12*c*: Integration in SAP Environment
DB: Option: Advanced Compression
2138262, Oracle Database 12*c* Advanced Compression for SAP Systems
2157904, Oracle 12*c*: Conversion of Compressed Tables
2166836, Oracle 12*c*: Problem during SAP Upgrade with Compressed Tables
2254836, BR*Tools Support for Oracle ADO/ILM
2254866, Using Oracle Database 12*c* Automatic Data Optimization
2255992, R3load and R3szchk: New Oracle Feature for Database ILM Policy
2258061, Enhancements for ADO/ILM for Table Conversions or System Copy
DB: Option: In Memory
2137032, DBA Cockpit: Monitor for In-Memory Feature
2178980, Using Oracle Database In-Memory with SAP NetWeaver based Products
2189163, Oracle Database In-Memory Advisor for SAP
DB: Option: Database Vault
2218115, Oracle Database Vault 12*c*

DB: Installation and Upgrade
1915299, Troubleshooting Software Installation for 12.1.0.2
1915301, Database Software 12.1.0.2 Installation on Unix
1915302, Database Software 12.1.0.2 Installation on Windows
1915315, Database Upgrade Scripts for 12.1.0.2
1915317, Migrating to Software Owner ‚oracle'
1915323, OS User Concept for Oracle Database 12*c* Release 1
2064206, Database Upgrade to 12.1.0.2 with Grid Infrastructure
DB: Patches
1915313, Current Patch Set for Oracle Database 12*c* Release 1 (12.1)
1915316, Database: Patches for 12.1.0.2
2145572, Grid Infrastructure: Patches for 12.1.0.2
DB: Instance Configuration
1888485, Database Parameters 12.1.0.2
DB: Admin: BR*Tools
2087004, BR*Tools Support for Oracle Database 12*c*
Engineered Systems
2145628, Exadata/SuperCluster: Patches for 12.1.0.2
2290084, SAP Software and Oracle Database Appliance Version 12.1