ORACLE
SUPERCLUSTER

Database Consolidation on Oracle SuperCluster

ORACLE WHITE PAPER | JULY 2015

ORACLE

Table of Contents

Introduction

As modern servers get more and more powerful with every generation, consolidation of multiple workloads on a shared infrastructure has become the driving force in reducing capital and operational expense in IT organizations. Consolidation not only improves resource utilization of otherwise underutilized servers, but also simplifies management and enables organizations to react to changes in demand more timely. Database consolidation is a central aspect of this effort, whether it is the consolidation of existing databases, new databases, or other forms of service delivery such as database-as-a-service (DBaaS) in cloud-based environments.

The key challenge in designing a consolidated infrastructure lies in the choice of the technologies. No single technology can best address all possible requirements at the same time. Therefore, a clear understanding of requirements and their relative importance, as well as the provided features and characteristics of potential deployment options, is essential for designing an infrastructure that best meets all goals.

Oracle provides numerous virtualization and consolidation technologies for databases that operate at different layers of the stack and allow organizations to design a database consolidation infrastructure according to their needs, ranging from mission-critical databases with highest isolation and availability requirements to highest density consolidation with on-demand resource allocation.

Oracle SuperCluster integrates these technologies in an engineered system designed for reliable, scalable, and efficient consolidation of databases and applications. This white paper discusses relevant design considerations and requirements as well as the different database consolidation options offered by Oracle SuperCluster. It provides guidelines and recommendations on how these best can be used and combined to build a consolidated database infrastructure.

The focus of this paper is to provide an overview and comparative evaluation of all database consolidation options on SuperCluster, each one rated with respect to its isolation, density, efficiency, elasticity, availability, and manageability aspects. Detailed descriptions of each of these options can be found in the referenced documents.

This paper assumes the reader is familiar with the Oracle databases including technologies such as Oracle Grid Infrastructure, Oracle Automatic Storage Management, Oracle Real Application Clusters

(Oracle RAC), and basic principles of the Oracle Solaris operating system. The References section provides further material about each of these technologies.

Overview of Database Consolidation Options

Database consolidation can be applied to the database server (by running multiple databases on the same server) or the storage (by storing data for multiple databases on the same storage). Server and storage consolidation are mostly independent of each other and are discussed separately in this paper.

Database Server Consolidation Options

Database server consolidation technologies can be classified into four different classes:

- Hardware Virtualization
- Operating System Virtualization
- Database Instance Consolidation
- In-Database Virtualization

Oracle SuperCluster offers the following technologies in each of these classes, which are introduced in the remainder of this section.

Hardware Virtualization	• Physical Domains (PDoms) • Logical Domains (LDoms)
Operating System Virtualization	• Oracle Solaris Zones
Database Instance Consolidation	• Instance Consolidation
In-Database Virtualization	• Pluggable Databases (PDBs) • Schema Consolidation

Physical Domains (PDoms)

Physical Domains (PDoms) is a feature only offered on Oracle SuperCluster M6-32. It allows the physical partitioning of a server into electronically isolated domains that behave just like physical servers, each with their own CPUs, memory, and I/O devices. PDoms provide the highest possible level of isolation where software errors or hardware failures do not propagate themselves beyond the domain in which they occurred.

The partitioning of resources is static and does not allow resources to be reassigned during runtime or shared among domains. A PDom therefore has the same characteristics as a physical server and is best suited for mission-critical databases that require the highest degree of isolation. It also can serve as a way to partition a machine into multiple smaller units, each of which then can be further partitioned into logical domains.

Logical Domains (LDoms)

An Oracle VM Server for SPARC domain, also referred to as a logical domain (LDom), is a virtual machine comprised of a discrete logical grouping of resources and is available on SPARC SuperCluster T4-4, Oracle SuperCluster T5-8, and Oracle SuperCluster M6-32. It is managed by a hypervisor that allows domains to be created, destroyed, reconfigured, and rebooted independently. Each domain runs its own operating system kernel and software stack, has its own identity, and can be managed independent of other domains.

Unlike other virtualization technologies, an LDom has exclusive access to the resources that are made available to it by the hypervisor, which enforces a strict partitioning of resources, while still allowing administrators to deassign resources from one domain and reassign them to another domain. LDoms therefore provide strong isolation between databases with dedicated CPU, memory, and I/O resources per domain. For Oracle software licensing, an LDom is considered a hard partition.

LDoms can serve different roles (the referenced documents provide details). The first domain of a server is always the control domain and serves administrative purposes like the configuration of other domains or the assignment of resources. At the same time, it is also an I/O domain. Applications are run either in I/O domains or guest domains. I/O domains own physical I/O devices, either a PCIe root complex, a PCIe device, or a single- root I/O virtualization (SR-IOV) function. These I/O devices have native performance without being mediated by a virtualization layer. Guest domains have only virtualized I/O devices. On SuperCluster, only I/O domains, which provide native I/O performance, are supported.

Every physical server and every PDom automatically have at least one LDom, also referred to as the primary domain, which can be assigned all physical resources of the server or PDom.

Oracle Solaris Zones

Oracle Solaris Zones is an OS-level virtualization technology provided by the Oracle Solaris operating system, and it allows the consolidation of multiple databases or applications on the same operating system using software-defined boundaries. Rather than virtualizing the underlying hardware, Oracle Solaris Zones virtualizes the operating system kernel and provides each database or application inside a zone a private execution environment without being able to see any resources (e.g., processes, files, and memory) of other zones while still sharing a common kernel. Inside an LDom, a global zone serves administrative purposes and provides resources to local zones.

Unlike LDoms, Oracle Solaris Zones does not enforce a hard partitioning of resources, but allows sharing of CPU, memory, and I/O among zones. Due to their lightweight nature, a deployment of a large number of zones within a system is possible. The I/O stack of zones is fully virtualized and provides close to bare-metal I/O performance. For Oracle software licensing, zones are considered a hard partition if configured with capped (dedicated) CPUs.

Oracle Solaris Zones also allows the running of older Oracle Solaris versions inside guest zones, referred to as *branded zones*. A global zone running Oracle Solaris 11 can host zones running either Oracle Solaris 10 or Oracle Solaris 11. A global zone running Oracle Solaris 10 can host zones running Oracle Solaris 8, Oracle Solaris 9, or Oracle Solaris10.

Oracle Solaris 11.2 introduced Oracle Solaris Kernel Zones, which run individual copies of the Oracle Solaris kernel. Neither Oracle Solaris 11.2 nor kernel zones are currently supported on SuperCluster, and kernel zones are therefore not described further in this paper.

Instance Consolidation

The deployment of multiple database instances on the same operating system is referred to as instance consolidation. All database instances share CPU, memory, and I/O devices as well as the operating system kernel

and Oracle Grid Infrastructure (including the Oracle Automatic Storage Management instance) for cluster and storage communication. Instances can be deployed either inside an LDom or Oracle Solaris Zones.

Database instances do not virtualize any resources, but directly access the resources provided by the operating system without any (further) virtualization layers. CPUs can be shared among instances either on demand, through physical partitioning of CPUs using resource pools, or by capping the maximum CPU consumption of an instance (instance caging).

In a clustered environment, Oracle Clusterware allows administrators to manage instances manually or through policies. For policy-managed databases, cluster nodes are assigned to server pools and automatically manage database instances within pools based on policies, which, for example, enable automatic restart of instances on another node in case of failures.

Pluggable Databases (PDBs)

Oracle Multitenant is a new option of Oracle Database 12*c*, in which pluggable databases (PDBs) are consolidated in a common container database (CDB). A CDB instance is a regular database instance, comprised of a system global area (SGA) and background processes, that is capable of hosting multiple virtualized databases called PDBs. PDBs share all resources of a CDB, including SGA memory for buffer cache and shared pool as well as background processes such as log writer, database writer, and parallel query processes.

Each PDB is a portable, self-contained database with its own schemas, schema objects (that is, tables and indexes), and non-schema objects (that is, users, roles, and tablespaces), stored in the individual PDB *system, sysaux,* and *user* tablespaces and associated PDB data files. A PDB can be plugged into a CDB, unplugged from a CDB, and opened or closed in a CDB instance independent of other PDBs. From an application's perspective, a PDB behaves just like an ordinary database. Since a PDB has virtually no static CPU or memory footprint, a CDB can efficiently host a large number of PDBs.

Schema Consolidation

A schema is a collection of logical structures of data, or schema objects. A schema is owned by a database user and has the same name as that user. Each user owns a single schema, and a database can have many schemas.

Similar to PDBs, schemas have no static CPU or memory footprint, but they lack the namespace and lifecycle isolation of PDBs. Schemas with identical names cannot be consolidated in the same database, and a schema cannot be opened or closed independent of other schemas.

Database Storage Consolidation Options

Oracle SuperCluster integrates Oracle Exadata Strorage Server, an intelligent scale-out storage optimized for database use. Oracle Exadata Storage Server provides the highest database performance by integrating smart flash cache and the capability of query offloading. Storage servers connect to database servers through high-speed InfiniBand are the preferred storage backend for database deployments on SuperCluster. Other supported storage solutions such as Oracle ZFS Storage Appliance or Fibre Channel-attached storage are not discussed in this paper.

In order to access Oracle Exadata Storage Server from a database node (both clustered or nonclustered), an installation of Oracle Grid Infrastructure is required, and it includes Oracle Clusterware and Oracle Automatic Storage Management. Each domain or zone requires its own installation of Oracle Grid Infrastructure, which is shared by all database instances, PDBs, or schemas running on this domain or zone.

Storage servers expose *grid disks* (disk partitions), which are created on *cell disks* (physical disks). Data files for tablespaces, voting files, and the Oracle Cluster Registry are stored in *disk groups*. Disk groups are created in Oracle Automatic Storage Management using a set of grid disks from multiple storage servers. Depending on the

selected redundancy level of *normal* or *high*, grid disks from three (normal) or five (high) different storage servers are required. Disk groups with normal redundancy can tolerate the loss of a single storage server (or mirror), while disk groups with high redundancy can tolerate even a loss of two storage servers (or mirrors).

Each *cluster* (including single-node clusters, single-node standalone installations, and true multinode clusters) requires its own disk groups on its own grid disks, which means that grid disks cannot be shared among clusters. Domains and zones that form different clusters require independent grid disks that cannot be shared among them. Storage planning is therefore an important task when consolidating databases, and it is described in detail in the documents referenced by this paper.

Dedicated Storage Server

For improved isolation, storage servers can be dedicated to one cluster only by using their grid disks in only one cluster. In this case, failures of one storage server affect only a single cluster, and storage performance for databases of one cluster is completely isolated from databases on another cluster. Also, disk failures or data corruption are isolated and do not affect any other cluster.

Such a deployment requires at least three or five storage servers per cluster, depending on the selected redundancy level, which limits the maximum number of clusters. This configuration is therefore mostly suitable for databases that require the highest storage isolation and are deployed in only a few large domains, or for deployments where many databases are deployed on the same cluster.

Shared Storage Servers

In consolidation scenarios with a larger number of clusters, storage servers can be shared among clusters. Since each cluster still requires its private grid disks, multiple grid disks (one per cluster) need to be created per cell disk. Using identical partitioning of all cell disks into the same number of grid disks simplifies management and ensures that physical disks are equally utilized. Proper sizing of grid disks requires thorough capacity planning. To allow a reserve for potential future growth, some amount of space per cell disks should be kept free.

Besides supporting a larger number of clusters, the advantage of a shared storage configuration is also a better and more balanced resource utilization. Since storage resources are dynamically shared among clusters, databases can potentially use a higher aggregate I/O bandwidth, more flash cache, and more CPU cycles for query offloading than in dedicated deployments. Performance effects between databases can be mitigated by using I/O Resource Management, a feature of Oracle Exadata.

Combination of Consolidation Options

The above-described server and storage consolidation options can be combined in many ways in order to find the best balance among various requirements. For example, a PDom can be partitioned into multiple LDoms, which each can host multiple Oracle Solaris Zones, each running multiple database instances hosting multiple PDBs or schemas. The following sections discuss each consolidation option in isolation, always seen from the consolidated database or tenant.

Database Server Consolidation Options

Consolidation using PDoms refers to systems that are partitioned into multiple PDoms, with a single database per PDom. Each PDom consists of a single LDom (the primary domain), which only hosts a single zone (the global zone) and runs a single database instance hosting only one database.

Consolidation using LDoms refers to systems (either physical servers or PDoms) that are partitioned into multiple LDoms. The primary domain only serves as a control domain. Databases are deployed in I/O domains, each consisting of only a single zone (the global zone) that runs a single database instance hosting only one database.

Consolidation using Oracle Solaris Zones refers to domains that host multiple zones. The global zone serves administrative purposes, while databases are deployed inside local zones. Each local zone runs a single database instance hosting only one database.

Instance consolidation refers to domains or zones that run multiple database instances on the same operating system, with each database instance hosting only one database.

Consolidation using PDBs refers to a multitenant CDB instance (inside a domain or zone) that hosts multiple PDBs.

Consolidation using schemas refers to a nonmultitenant database (nonCDB) instance (inside a domain or zone) that hosts multiple schemas.

Database Storage Consolidation Options

A dedicated storage approach is possible only if the number of clusters is low, which is typically the case if databases are consolidated in large domains. If multiple databases are deployed on the same cluster, as is the case with instance consolidation, PDBs, or schema consolidation, they all share storage servers but can be isolated from another cluster.

Instances consolidated on the same cluster could theoretically still each use dedicated storage servers. However, since instances already share the domain and OS, storage isolation leads to hardly any advantage in this case.

A shared storage approach is applicable to all database server consolidation options and yields better resource utilization than dedicated storage servers.

Consolidation Requirements and Considerations

When multiple databases are consolidated on a common set of physical resources, special attention needs to be paid to how these resources are allocated to databases or shared among them. Typically these decisions lead to a trade-off between isolation and density. Organizations need to understand these trade-offs and carefully decide based on their requirements and priorities how best to balance them.

This section classifies requirements into six categories (isolation, density, efficiency, elasticity, availability, and manageability) and gives a brief introduction to each of them. The following section then evaluates all described consolidation options based on these requirements and discusses their advantages and disadvantages with respect to each requirement in detail.

Physical resources of interest are mainly CPU, memory, I/O devices and bandwidth on the server side, and storage capacity, flash cache, and CPU on the storage side.

Isolation

Before the rise of consolidation, databases were fully isolated, each deployed on their own server, running their own operating system and database software, and connected to their own storage through dedicated I/O interfaces. When consolidating databases or implementing DBaaS solutions, organizations need to decide how much isolation they must maintain, and how much they can afford to give up. Compromising on isolation attributes that were taken for granted in the past may be an unpleasant decision to make, but isolation comes at a cost. By giving up some degree of isolation, organizations may be able to greatly increase the density, efficiency, and elasticity of their database infrastructure and significantly improve manageability.

Databases can be isolated from each other with respect to lifecycle, namespace, security, faults, and performance.

Lifecycle isolation describes the ability for databases to maintain an independent lifecycle—to run independent software versions and patch levels, have their maintenance operations (that is, security patch installations) decoupled, and be started, stopped, opened, or closed independent of other databases.

Namespace isolation allows databases the free use of names such as user names, service names, or host names without colliding with other consolidated databases.

Security isolation ensures that one database cannot accidentally or maliciously access or manipulate data of another database.

Fault isolation keeps software errors or hardware faults local to the database, operating system, domain, or storage in which they occur without affecting any other databases.

Errors and faults include I/O errors, process or instance crashes, kernel panics, and any kind of hardware failures.

Performance isolation aims at eliminating any interference that one database may have on the performance of another database—that is, throughput, response times, or other quality- of-service (QoS) metrics.

Density

Reduction of cost is one of the key drivers in database consolidation. The density of a deployment is measured by the number of supported databases on a given set of hardware resources, which directly affects capital expenditures (CapEx). Organizations therefore typically try to maximize the density of their deployments without violating any other constraints (that is, isolation requirements) in order to minimize their CapEx.

The density of a consolidated database deployment can be improved in two ways: by reducing the footprint per database and by allowing oversubscription of resources.

The *footprint* of a database is comprised mainly of the allocated disk space for software (that is, operating system and database software) and data (that is, data files for tablespaces) and the amount of consumed memory (that is, OS kernel, database SGA and PGA, and process memory such as heaps and stacks). In some cases, also CPU resources, that are required for background tasks unrelated to load (that is, monitoring threads and processes) can have a nontrivial impact on the footprint of a database. The footprint of a database can be reduced through sharing of these resources; for example, by running multiple databases on the same OS kernel.

A deployment is referred to as *oversubscribed* if the aggregate peak resource demand of databases exceeds the amount of physical resources. Oversubscription is accomplished through resource sharing, which allows a database to use resources (that is, CPU or memory) that are currently unused by other databases. Oversubscription requires the ability to dynamically assign resources based on demand, and therefore conflicts with static resource allocations.

Efficiency

Efficiency is a metric that describes how well a workload is executed on a given environment and determines its aggregate performance. Efficiency manifests itself in either cost (overhead) or savings (gains). By comparing the performance of a consolidated deployment against a nonconsolidated deployment, the efficiency of a consolidation technology can be determined.

The efficiency *cost* of consolidation is usually related to virtualization overhead. Savings can result from deduplication of common functionality; for example, the sharing of background processes in case of PDBs or schema consolidation.

Elasticity

In nonconsolidated environments, hardware resources are often provisioned for peak load and only fully utilized for a few hours per day, or even few days per year. The majority of the time, resources are underutilized. Consolidation aims at reducing capital expenditures by increasing average resource utilization of the infrastructure through sharing of resources among multiple databases. In such environments, the elasticity of the infrastructure describes the ability to dynamically allocate resources to databases as needed. The better the infrastructure can react to changes in demand and provide resources as needed in a timely manner, the higher the density it supports (provided workloads do not all peak at the same time).

Elasticity is facilitated by *dynamic resource allocations within a server*; for example, the allocation of CPU cycles to the database with highest demand, eventually based on policies and priorities.

If the aggregate resource consumption of all databases exceeds the physical resources of a server, *load balancing across servers* becomes necessary. The ability to also manage resources across servers, often implemented as *live migration*, further improves the elasticity of a deployment.

Availability

Many applications serve important business functionality and are required to be continuously available. The availability of the database backing such applications, that is the percentage of time it services requests (*uptime*), is therefore an equally important metric in consolidated environments as in nonconsolidated environments. A database needs to be available both in cases of unexpected failures as well as during planned maintenance operations.

During the consolidation of many databases on a common infrastructure, hardware or software failures of one component may affect more databases than in fully isolated deployments. *Fault isolation* is therefore an important

consideration for availability. The *fault recovery* time describes how quickly a database can recover from a fault. To maintain availability even in the event of faults, organizations can implement redundancy through clustering or *replication of databases*. Also, *maintenance operations*, such as operating system or database updates and patch installations, may affect the availability of a database.

Manageability

While capital expenditures are much easier to assess than operational expenditures, the costs for operating and managing a database deployment are at least equally important to consider, and may even exceed the CapEx costs. Manageability may vary greatly depending on the chosen deployment model. The possibility to reduce the number of entities to manage, or to *manage many as one*, can significantly reduce OpEx costs.

Installation, updating, and patching of operating system and database software are tasks that can be simplified in consolidated environments through the use of images and templates, or by sharing a common OS or database installation among multiple databases.

The number of steps needed to *provision new databases* in a shared environment is an important consideration, especially in DBaaS deployments, where provisioning of new tenants can be a frequent task.

To protect against loss of data, organizations deploy *backup and restore* as well as *disaster recovery* solutions. Database consolidation can potentially reduce the management costs for such solutions if many databases can be backed up or replicated as one.

Implementing Database Consolidation

No size fits all, and database consolidation is no different. Isolation and density are conflicting goals, and each consolidation option has different strengths and weaknesses. This sections discusses the advantages and disadvantages of all database consolidation options on Oracle SuperCluster and rates them with respect to the aforementioned requirements. It also provides general guidelines and recommendations.

General Guidelines

With respect to the design of a consolidated database infrastructure, the key challenge lies in finding the best compromise between sometimes conflicting requirements. While PDoms provide the strongest isolation, pluggable databases achieve the highest density; and LDoms, zones, and instance consolidation range in between. Any deployment based on only one of these technologies will almost certainly be biased towards only a few of the requirements. In most cases, the ideal deployment that best balances all requirements is one that leverages a combination of multiple technologies. While it is in principle possible to run hundreds of database instances on the same OS, to encapsulate each instance inside a zone and deploy hundreds of zones, or to provision hundreds of PDBs in a single CDB, such deployments are rarely the ones that combine isolation, density, and efficiency in the best possible way.

Even though the remainder of this section discusses each of the consolidation technologies by itself, the reader should be aware that a combination of multiple technologies is possible and strongly advised. At the end of this section, this paper provides recommendations on each of the technologies as well as their best combination.

Ratings

All consolidation options are rated using one through five stars (★) with respect to each requirement. The more stars, the better they fulfill a given requirement. An empty star (☆) is used when a deployment option does not support a certain requirement.

Evaluation of Database Server Consolidation Options

The term *cluster* is used for multinode database clusters as well as single-node clusters and standalone installations of Oracle Grid Infrastructure.

The term *socket* refers to a *processor* with its associated memory, PCIe root complex, and PCIe devices. A processor has multiple cores. The *strands* or *threads* of a core are referred to as *CPUs*. For example, Oracle's SPARC T5 is a *processor* with *16 cores* and *8 CPUs* per core (a total of 128 CPUs per processor).

Isolation

Databases in a consolidated environment can be isolated from each other with respect to lifecycle, namespace, security, faults, and performance.

Lifecycle

PDoms have a completely decoupled lifecycle and can be power-cycled, started, stopped, installed, upgraded, and patched completely independent of other PDoms and run individual OS and database software versions.

LDoms have decoupled lifecycles just like PDoms with the exception of not being able to be power-cycled individually. However, LDoms still can be rebooted independent of each other. If a domain does not own an entire PCIe root complex, but only a PCIe device or SR-IOV function, it will be affected by reboots of the domain that owns the root complex (that is, the primary domain).

Zones can be started and stopped independent of each other. While branded zones allow the use of older Oracle Solaris versions inside a zone, they still share the kernel of the global zone and all are updated when the global zone is updated. Branded zones allow the use of different Oracle Solaris versions, but not different patch levels of the same version. The database software is installed in each zone individually. While zones have a dependency on the OS version and patch level of the global zone, database versions and patch levels within a zone are entirely independent of those in another zone.

Consolidated database instances share the same OS and therefore all use the same OS version and patch level, and all are affected by reboots of the operating system. They also share a common Oracle Grid Infrastructure installation, and all are affected if the cluster is updated or stopped. While it is possible to install different versions of the Oracle Database software in multiple Oracle homes on the same OS, such a deployment may not be the best approach. If a better lifecycle isolation between databases is required, encapsulation of instances in zones should be considered instead. With instance consolidation, all database instances will run the same database software version and patch level, but they can be individually started and stopped and can have individual initialization parameters.

PDBs share a common container database (CDB) and operating system. They all run on the same operating system, Oracle Grid Infrastructure, Oracle Database version, and patch level and are affected by OS reboots and CDB restarts. Some initialization parameters can be configured on a per-PDB basis, while others are global for the CDB. Within a CDB, PDBs can be plugged, unplugged, opened, or closed individually.

Schema consolidation leads to a tightly coupled lifecycle of all schemas deployed in the same database. Operating system, cluster, database version, database instance, and initialization parameters all are shared among schemas. Unlike PDBs, schemas cannot be opened or closed individually, but all depend on the hosting database instance.

Namespace

PDoms, LDoms, and zones allow completely isolated namespaces between databases. Each domain or zone has its own host names and OS user names and allows databases to freely use service and schema names.

Instance consolidation and PDBs require all databases to share the same host name and OS user name (unless database instances use multiple Oracle homes). Database and service names must be unique but can otherwise be chosen independently. Each database can have multiple schemas with isolated namespaces for schema names.

Schema consolidation only isolates the namespace for schema objects such as tables and indexes. Multiple schemas cannot use the same schema name, and all use the same host name, database name, and service name(s).

Security

Partitioning of resources in PDoms and LDoms allows these deployments to fulfill highest security requirements in which databases in one domain can access no data in another domain.

Oracle Solaris Zones isolates processes, file systems, and memory between zones and prevents a zone from accessing data in another zone. Since zones share a common kernel, the provided isolation is slightly lower than with PDoms or LDoms.

Database instances sharing a common OS typically run as the same OS user, typically named “oracle”. Each database instance therefore has access to another database's data in the file system or on Oracle Automatic Storage Management disk groups, which could be exploited by malicious users who gain access to the operating system. The same applies to PDBs and schemas. For instance consolidation, it is possible to enhance security by installing the database software in multiple Oracle homes, each using their own UNIX user and group IDs. While this improves security isolation between databases, it also increases management complexity and imposes a cost at runtime due to increased instruction cache misses resulting from the use of multiple copies of the Oracle binary. If additional security is required, the use of zones should be considered instead.

Since PDBs and schemas additionally share the database instance itself, they attach to a common SGA, use the same UNDO tablespace and write to the same REDO logs. Privileges and roles can restrict users so they cannot access OS resources or data in other PDBs or schemas, but the operating system itself cannot prevent any of these accesses.

Faults

PDoms are electronically isolated, and faults in one domain do not affect any other domains unless the entire rack, server room, or building is affected (as in the case of disasters).

Software faults (e.g., kernel panics or instance crashes) in one LDom never propagate themselves to another LDom. If all resources of a socket (that is, CPUs, memory, and PCIe root complex) are assigned to only one LDom, a failure of any of these resources does not affect any other LDom. If two LDoms share hardware—for example, an I/O device, a memory channel or dual inline memory module (DIMM), or CPUs from the same socket—a hardware failure might affect multiple domains. If a domain does not own an entire PCIe root complex, but only a PCIe device or SR-IOV function, it will be affected by faults of the domain that owns the root complex (that is, the primary domain).

Zones share not only common hardware, but also the OS kernel. Hardware failures in the domain or failures of the global zone affect all zones. However, software failures like instance crashes or cluster failures within a zone do not impact other zones.

Consolidated Instances share hardware, operating system, and a common cluster. Any failures of the OS or cluster node affect all instances. Crashes of a single instance do not impact other instances.

PDBs and schemas also share a common database instance, and all are additionally affected by a database instance failure.

Performance

Databases are completely isolated with respect to performance as long as they do not share any hardware components, as is the case with PDoms (unless storage servers are shared among PDoms).

While LDoms also partition resources, special attention must be paid to how partitioning is done. As long as LDoms are partitioned along socket boundaries with all CPUs, memory, PCIe root complex, and PCIe devices from one socket only assigned to one LDom, performance between LDoms is completely isolated. If, however, LDoms share CPUs from the same processor, they also share the last-level cache of the processor, which can lead to mutual performance effects between domains.

Zones share memory and memory controllers, and potentially also CPUs. Uncapped zones without explicit CPUs assigned compete for CPU cycles with other zones. Since fair-share scheduling (FSS) is not supported on SuperCluster for databases in zones, no CPU resource management can be implemented for uncapped zones. Capped zones with dedicated CPUs cannot steal CPU cycles from another zone. However, if zones share CPUs from the same processor or core, different zones still share hardware caches and potentially other processor components such as integer pipelines, which creates performance dependencies between them. To reduce mutual performance effects, capped zones should be assigned CPUs along core boundaries.

Database instances share not only memory, but also I/O devices and potentially CPUs. Without further administration, database instances compete with other instances for CPU cycles. Instance caging restricts the maximum amount of consumed foreground CPU of a database and therefore limits its maximum CPU demand. However, caged instances still run on all available CPUs and share hardware caches, pipelines, and other processor components. Instance caging also does not restrict the CPU consumption of background processes of an instance. To reduce interference, database instances can be assigned to resource pools, which allows partitioning of CPUs among instances. Similar to zones, CPUs should be allocated along core boundaries to reduce mutual effects due to sharing of hardware components.

PDBs and schemas share not only CPU and I/O devices, but also SGA memory (that is, buffer cache and shared pool) and background processes. The Database Resource Manager allows to configure CPU shares or caps per PDB or user through the configuration of resource plans. While this can reduce mutual performance effects, CPUs cannot be partitioned among PDBs or schemas, which leads to sharing of processor components such as caches. PDBs and schemas also compete for buffer cache, and all undergo the same replacement algorithms of common default, keep, and recycle pools. They also share parallel query slaves, log writer, and database writer processes. An increase in the amount of generated redo data in one PDB or schema can, therefore, also affect other tenants.

Rating

Isolation	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Lifecycle	★★★★★	★★★★★	★★★★	★★★	★★	☆
Namespace	★★★★★	★★★★★	★★★★★	★★★	★★★	★
Security	★★★★★	★★★★★	★★★★	★★	★★	★
Faults	★★★★★	★★★★	★★★	★★	★	★
Performance	★★★★★	★★★★	★★★	★★	★	★

Density

The maximum number of supported databases on a given set of physical resources is mainly affected by the footprint per database and the ability to oversubscribe resources.

Footprint

PDoms and LDoms require the running of a dedicated Oracle Solaris kernel, and therefore have a much higher memory footprint than other consolidation options.

Zones share a common Oracle Solaris kernel, but (like PDoms and LDoms) require an individual Oracle Grid Infrastructure stack running an Oracle Automatic Storage Management instance and Oracle Clusterware processes inside each zone in addition to the database instance itself. They also require exclusive Oracle Automatic Storage Management grid disks for Oracle Cluster Registry, a feature of Oracle RAC, and disk groups managed by this cluster.

Instance consolidation allows the sharing of the cluster among multiple databases. It reduces the footprint of a database to SGA, private memory, and background processes of the database instance (including the CPU consumption of background processes). With many consolidated instances, their memory and CPU footprint still can be significant. However, since they all share a common cluster, they can share disk groups (and grid disks), which significantly reduces the storage footprint of a database.

PDBs and schemas also share SGA memory and background processes of a common database instance and have virtually no memory or CPU footprint. Since they also share UNDO tablespace and REDO logs, their storage footprint is reduced to their individual tablespaces and data files.

Oversubscription

PDoms and LDoms partition resources among domains and allow no sharing of CPU and memory, and therefore no oversubscription of these resources.

Uncapped zones share CPU cycles on demand and therefore allow oversubscription of CPU. Capped zones are assigned dedicated CPUs and in consequence allow no CPU oversubscription. Even though zones share memory, the majority of a database's memory is comprised of its SGA, which is allocated at instance startup and not freed automatically when memory pressure increases. Any oversubscription of memory leads to paging when physical memory becomes low, which can lead to significant performance loss and system instability. Deployments of zones therefore practically speaking, do not allow oversubscription of memory.

Like zones, consolidated database instances allow sharing of CPU if they are not bound to a resource pool. When bound to a resource pool, CPUs cannot be shared with other databases outside the pool—which limits oversubscription but still may allow some oversubscription if multiple database instances are bound to the same resource pool. For the same reasons as for zones, memory oversubscription is not possible.

PDBs and schemas are the only deployment models that allow both oversubscription of CPU and memory since neither of these resources is partitioned or statically allocated to a tenant. CPU cycles are shared on demand (and eventually managed through Database Resource Management) among PDBs and users. Each PDB or user can dynamically use SGA memory (e.g., buffer cache using default cache replacement algorithms). Also, PGA and other private memory can be allocated on demand but should never grow beyond the physical resources of the server.

Rating

Density	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Footprint	★	★	★★	★★★	★★★★★	★★★★★
Oversubscription	☆	☆	★★★	★★★	★★★★★	★★★★★

Efficiency

The efficiency of executing a workload depends both on cost (overhead) and potential savings (gains).

Cost

PDoms deliver bare-metal performance and impose absolutely no overhead on a workload. The use of PDoms adds no cost compared to nonconsolidated deployments.

Also, LDoms deliver bare-metal performance, and zones only add a small overhead for virtualized I/O devices. While the immediate overhead resulting from these technologies is nonexistent (LDoms) or small (zones), both LDoms and zones require a private installation of the Oracle software (individual Oracle home) which might affect cost if LDoms or zones share a processor and its caches. At runtime, multiple copies of the program code (in same or different versions) can increase pressure on shared hardware instruction caches, and this increases with the number of Oracle homes and can degrade performance. This cost can be avoided if LDoms and zones are deployed along socket boundaries, which often is the case for LDoms.

Database instance consolidation and schemas impose no overhead at all, and PDBs nearly no overhead. Only for DML-intensive workloads, PDBs add a minimal cost for switching between the PDB and root container of the CDB, which is typically more than amortized for by the efficiency gains of PDBs.

Savings

Since neither PDoms or LDoms eliminate any common functionality of multiple databases compared to nonconsolidated deployments, they also do not lead to any efficiency gains.

Zones could theoretically create small gains by sharing a common OS kernel among databases, which eliminates duplicate work of some kernel threads. However, the amount of CPU time spent in the database exceeds by far the time for background activities in the OS, so that zones practically do not lead to improved workload efficiency.

Similar to zones, instance consolidation could theoretically have efficiency gains resulting from savings of common kernel or cluster tasks. Practically, those savings are minimal.

PDBs and schemas share a common database instance and can benefit greatly from the elimination of duplicate work performed in background processes. In particular, the aggregation of work in database writer processes (when flushing dirty database buffers to disk) and log writer processes (when writing redo log to disk) can lead to significant savings and improved efficiency of PDB and schema-based deployments.

Rating

Efficiency	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Low Cost	★★★★★	★★★★	★★★	★★★★★	★★★★★	★★★★★
Savings	☆	☆	★	★	★★★★★	★★★★★

Elasticity

The elasticity of an environment describes the ability to reassign resources on demand.

Resource Allocation

Both PDoms and LDoms partition resources and therefore do not allow any dynamic resource allocations if the demand of a workload changes. However, LDoms support reconfiguration during runtime, which allows administrators to deassign CPUs or memory from one domain and assign it to another without rebooting the domains. A database inside the domain can make immediate use of additional CPU resources. Memory reassignments can be disruptive and might require database instance restarts.

Databases inside uncapped zones can dynamically share CPU cycles. However, CPU allocations to zones cannot be controlled or managed (*described on page 15*).

Consolidated database instances can dynamically share CPU cycles if not bound to a resource pool. Instance caging caps the maximum amount of CPU per database (*described on page 15*).

Due to the mostly static allocation of SGA memory, only PGA memory can be dynamically allocated for databases inside zones or instance consolidation.

PDBs and schemas can allocate CPU cycles as needed. The amount of CPU cycles consumed by sessions accessing a PDB or schema can be controlled through resource management (*described on page 15*). They share all memory structures of a common database instance and can dynamically allocate buffer cache, shared pool, and memory from other pools as needed. This allows PDBs and schemas to efficiently react to changes in workload behavior and satisfy the CPU and memory resource needs of a tenant if sufficient resources are available.

Live Migration

PDoms and zones do not support live migration to another server. LDoms support live migration in general, but not for InfiniBand-based protocols that are used between database nodes and storage servers and for cluster communication between Oracle RAC nodes. Databases inside LDoms therefore cannot be migrated live on Oracle SuperCluster.

Consolidated Oracle RAC database instances can be *migrated* within the cluster by starting a new instance on another cluster node and then stopping the original instance. This also applies to Oracle Real Application Clusters One Node databases. The use of policy-managed databases further simplifies and automates this procedure. Except for a short period during instance reconfiguration, the database instances keep servicing requests throughout this process. Clients may use connection pools like the Oracle Universal Connection Pool to detect an instance startup and shutdown and automatically reconnect or load-balance accordingly. Application Continuity, a feature introduced with Oracle Database 12*c*, can further mask a short service interruption by recovering inflight transactions.

A PDB can be seamlessly migrated between Oracle RAC nodes of a clustered CDB by relocating its services to the target node (which implicitly opens the PDB there), and then closing it on the original node. With the use of the Oracle Universal Connection Pool, this migration is completely transparent to the application, absolutely free of downtime or failed requests, and imposes only a minimal degradation of response times on the PDB during migration. A document describing this procedure in detail can be found in the References section.

Schemas are not associated with services and therefore cannot be migrated like PDBs. Theoretically, an administrator could define one service per schema and implement a similar migration as described for PDBs. However, schemas cannot be opened or closed individually. Access to a schema on a particular node can therefore not be safely restricted, which does not allow to safely restrict access to a schema to certain nodes.

The migration of an Oracle RAC instance or PDB frees all CPU and memory resources that it originally consumed on the node it migrated off. While the term *migration* is often associated with migration of virtual machines (VMs), the migration described here is only implemented at a different layer of the stack. It has all the attributes typically associated with live migration and therefore qualifies as such.

Rating

Elasticity	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Resource Allocation	☆	★	★★★	★★★	★★★★★	★★★★★
Live Migration	☆	☆	☆	★★★	★★★★★	★

Availability

The availability of a database depends on the probability of its being affected by a fault, the time to recover from a fault, the degree of redundancy in hardware and software, and the amount of downtime imposed by maintenance procedures.

Fault Isolation

The isolation of hardware and software faults already has been discussed in the Isolation section. All conclusions from that section also apply here.

Fault Recovery

The time to recover from a fault depends primarily on the time to restart a database instance (potentially including a reboot of the operating system) or to fail over to another database instance on a different server. Besides the implemented redundancy option, which can be chosen independent of the consolidation model and is therefore not further discussed here, the fault recovery time depends mostly on the size of the failed database instance.

If multiple tenants are consolidated in larger database instances (as in the case of PDBs and schema consolidation), those databases have typically a larger SGA and run higher transaction rates than individual single-tenant databases deployed on the same OS (instance consolidation) or inside zones or domains. The allocation of SGA memory and the recovery of transactions in case of a failure may therefore take longer in case of PDBs and schema consolidation than for other consolidation options.

If redundancy is implemented, failover times mostly depend on the chosen redundancy option and can be as short as a few seconds with Oracle RAC. Larger database instances might experience longer recovery times if instance restarts are required or a larger number of transactions needs to be recovered.

Clustering and Replication

While fault isolation may limit the impact of a fault on only a subset of databases, none of the described consolidation models can prevent faults from happening. In order to maintain availability even in the event of faults, database deployments must implement redundancy to protect against such failures. In a redundant configuration, database service can be resumed on another set of resources (that is, another server, domain, or zone) in case of database instance, operating system, or hardware component failures.

Oracle offers a variety of products to implement redundant database configurations using both active-active as well as active-passive configurations.

Oracle Real Application Clusters (Oracle RAC) protects against failures such as a server failure, an operating system panic, or a database instance crash, and it supports both active-active as well as active-passive

configurations. An Oracle RAC database can be opened and simultaneously accessed from multiple database instances running on different nodes of a cluster. Oracle Real Application Clusters One Node allows a single Oracle RAC instance to be restarted automatically on another cluster node in case of failures. Each cluster node should reside on a different physical server or a different PDom to guarantee that any single failure (including hardware and server failures) only impacts a single node of the cluster.

Data Guard, a feature of Oracle Database, and Oracle GoldenGate allow a database to be replicated within or across data centers and support both active-active as well as active-passive configurations. Similar to Oracle RAC, the replicated database should reside on a different physical server or PDom, or in a different data center for geographic redundancy.

Oracle RAC, Data Guard, and Oracle GoldenGate can be used with any of the described consolidation options and apply in the same way to all of them.

Availability During Maintenance

Patch installations, software upgrades, and hardware repair often require that a database instance is temporarily shut down. When using Oracle RAC, a database can be simultaneously accessed by multiple database instances on different nodes. When one database instance is shut down, other database instances on other nodes continue to serve requests, and in this way, overall availability is maintained. By shutting down one database instance, applying patches to that node, starting it up again, and then repeating this process with the remaining nodes, an entire cluster can be patched or updated without downtime. This process is often referred to as a *rolling update*.

Major upgrades or certain patches might sometimes require the entire cluster to be shut down. In these cases, Data Guard or Oracle GoldenGate can help to minimize overall downtime by temporarily switching over to a standby database or cluster.

These technologies are applicable to all database consolidation options on SuperCluster in the same way. Pluggable databases can help to further minimize downtime of nonrollable upgrades.After the target software version is installed and a new CDB instance is brought up on the same cluster, a PDB can be unplugged from the old CDB and plugged into the new CDB. Since both CDBs share the same storage, these operations do not need to copy any data and are therefore very fast. After all PDBs have been unplugged from the old CDB and plugged into the new CDB, the old CDB can be destroyed.

Rating

Availability	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Fault Isolation	★★★★★	★★★★	★★★	★★	★	★
Fault Recovery	★★★★★	★★★★★	★★★★★	★★★★★	★★★	★★★
Clustering/Replication	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★
Maintenance	★★★★	★★★★	★★★★	★★★★	★★★★★	★★★★

Manageability

The manageability of a database deployment depends primarily on the amount of effort needed to install, update, and patch the software, provision new databases, and implement backup and recovery solutions.

Installation, Updating, and Patching

The effort of software maintenance such as installation, updating, and patching increases with the number of entities to manage (that is, operating system and database installations).

PDoms and LDoms require private installations of the operating system, Oracle Grid Infrastructure, and database software. Updates to any of these software components need to be applied to each one individually.

Deployments using zones, instance consolidation, PDBs, and schemas reduce management complexity trhough the sharing of a common operating system installation among multiple databases. OS updates and patches therefore only need to be installed once.

Instance consolidation, PDBs, and schemas further allow the sharing of a common Oracle Grid Infrastructure and database software, which further reduces the effort of maintaining many database installations.

Provisioning of Databases

If additional databases need to be provisioned on a shared infrastructure, resources must be allocated to these databases. For deployment models that partition resources, such as PDoms and LDoms, this may become very difficult if resources first have to be deallocated from existing domains (only possible with LDoms). For these deployment models, it is therefore better to leave resources unassigned for potential future growth.

The provisioning of new databases inside zones is easily possible without changes to existing zones if sufficient physical memory is available. Resource pool sizes can be dynamically changed in case of capped zones.

For PDoms, LDoms, and zones, the provisioning of new databases requires the creation of new grid disks on storage servers. It is therefore essential that enough free space is left on cell disks during the design of the initial storage layout.

Instance consolidation allows newly created databases to use existing disk groups, which greatly simplifies storage space allocation. As long as sufficient physical memory is available, the creation of a new database does not require any changes to existing resources with the potential exception of adjustments of resource pools or instance caging.

PDBs and schemas have no static resource allocation besides disk space in existing disk groups. The creation of new PDBs and schemas therefore requires no resource adjustments to any other tenants.

PDBs allow cloning of databases from existing templates, which can accelerate the time to provide a new database to a tenant compared to other consolidation options.

Backup, Restore, and Disaster Recovery

The complexity of backup, restore, and disaster recovery (DR) solutions depends on the number of database instances to manage. For PDoms, LDoms, zones, and instance consolidation, each database needs to be managed independently.

PDBs and schemas share a common database. A backup, restore, or replication of this database automatically covers all hosted PDBs or schemas and reduces administrative effort by managing many databases as one.

Rating

Manageability	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Install and Update	★	★	★★★	★★★★★	★★★★★	★★★★★
Provisioning	★	★★	★★★	★★★★	★★★★★	★★★★
Backup and DR	★★★	★★★	★★★	★★★★★	★★★★★	★★★★★

Overall Rating

	PDoms	LDoms	Zones	Instances	PDBs	Schemas
Isolation						
Lifecycle	★★★★★	★★★★★	★★★★	★★★	★★	☆
Namespace	★★★★★	★★★★★	★★★★★	★★★	★★★	★
Security	★★★★★	★★★★★	★★★★	★★	★★	★
Faults	★★★★★	★★★★	★★★	★★	★	★
Performance	★★★★★	★★★★	★★★	★★	★	★
Density						
Footprint	★	★	★★	★★★	★★★★★	★★★★★
Oversubscription	☆	☆	★★★	★★★	★★★★★	★★★★★
Efficiency						
Low Cost	★★★★★	★★★★	★★★	★★★★★	★★★★★	★★★★★
Savings	☆	☆	★	★	★★★★★	★★★★★
Elasticity						
Resource Allocation	☆	★	★★★	★★★	★★★★★	★★★★★
Live Migration	☆	☆	☆	★★★	★★★★★	★
Availability						
Fault Isolation	★★★★★	★★★★	★★★	★★	★	★
Fault Recovery	★★★★★	★★★★★	★★★★★	★★★★★	★★★	★★★
Redundancy Options	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★
Maintenance	★★★★	★★★★	★★★★	★★★★	★★★★★	★★★★
Manageability						
Install and Update	★	★	★★★	★★★★★	★★★★★	★★★★★
Provisioning	★	★★	★★★	★★★★	★★★★★	★★★★
Backup and DR	★★★	★★★	★★★	★★★★★	★★★★★	★★★★★

Recommendations

As the evaluation shows, strong isolation requires sacrifices in density, efficiency, elasticity, and manageability. High availability can be achieved through redundancy regardless of isolation. This section provides general recommendations on how to best leverage the presented consolidation options to create a scalable, secure, efficient, and reliable infrastructure for database consolidation or database-as-a-service offerings.

1. **Combine consolidation options to best balance requirements.** Individual requirements can be best achieved by a combination of the presented consolidation options. However, a nested deployment that leverages all of them might not be the best choice either. Instead, a combination of two or three technologies potentially can yield best results without increasing deployment complexity.
2. **Isolate where necessary, share where possible**. Fully isolated deployments cannot leverage many of the advantages of consolidation and can become difficult to administer when a large number of isolated entities needs to be managed. However, isolation needs often can be combined with high density, efficiency, and elasticity.

Organizations should carefully decide where isolation is required, and try to group databases by isolation needs. For example, mission-critical databases can be kept isolated from less critical databases. Databases of different departments that require security isolation can be isolated, while databases of the same department can share resources for improved density and efficiency.

3. **Avoid the extremes**. Hundreds of database instances running on the same operating system all will fail if the operating system fails. Hundreds of small domains or zones not only limit efficiency, but more important, increase the number of entities that need to be managed. Hundreds of PDBs or schemas in a single database instance can be efficient, but potentially require tuning effort to scale up the instance and increase mutual performance impact—and all are affected by an instance crash.
4. **Keep the number of domains and zones to a minimum**. Each domain or zone requires a dedicated Oracle Grid Infrastructure and database installation. Domains additionally require a dedicated OS installation. The creation of many domains or zones adds management complexity as the number of entities increases. Since each cluster requires dedicated Oracle Automatic Storage Management disk groups, storage resources must be carved out upfront and they require thorough planning. Partitioning of memory, and potentially CPU resources, limits sharing and therefore the possibility of oversubscription and on-demand resource allocations.
5. **Implement clustering or replication to achieve high availability**. Isolation only isolates faults or the impact of maintenance activities but does not prevent them. Regardless of the chosen consolidation option, additional preventive action must be taken to maintain availability in the event of failures. Oracle RAC, Data Guard, and Oracle GoldenGate allow clustering or replication of databases in active-active or active-passive configurations for all presented consolidation options. Two nodes of a cluster, or source and target database for replication, should always be deployed on two different physical servers or inside two PDoms to protect against not just software but also hardware failures.
6. **Use PDoms to create fully isolated domains.** Each consolidated database typically only requires a fraction of the physical resources of a server. On Oracle SuperCluster M6-32, PDoms allow the creation of electronically isolated domains that can be used to build highly available clusters with one cluster node each per PDom.
7. **Optimize isolation and performance by deploying LDoms along socket boundaries**. Mutual performance impacts among LDoms can be completely eliminated by assigning all resources of a socket, including its memory and PCIe root complex, to a single LDom only. By using one to four sockets per LDom, a server or PDom can be partitioned into smaller strongly isolated units that each run their own operating system kernel without adding any performance overhead. They further isolate operating system failures and allow to run different software versions and patch levels in different domains.
8. **Use Oracle Solaris Zones for security isolation**. Zones are lightweight since they share a common kernel with the global zone, which make zones the preferred technology for isolation on a subsocket level. While they provide only little fault isolation, they create a strong security boundary and can be used for security isolation between databases. If security isolation is required, zones are preferred over instance consolidation with multiple UNIX user IDs. Since zones each have their own Oracle Grid Infrastructure and database installation, they also allow the use of different database versions and patch levels and further isolate cluster failures.
9. **Minimize the number of sub-socket LDoms and zones**. While LDoms and zones do not have any performance overhead by themselves, the installation of multiple copies of the database software reduces efficiency at runtime due to cache misses if hardware caches of a processor are shared among domains or zones. It is recommended that administrators deploy no more than one LDom or zone per core. For performance-critical databases, the preferred approach is to use two to four cores per LDom or zone. With very small domains or zones, enough free memory must be kept to ensure stable operation and SGA memory should not be oversized.
10. **Consolidate database instances in the same domain or zone whenever possible**. If isolation requirements permit, instance consolidation is preferred over encapsulation of single instances inside domains or zones. Consolidated database instances run on a common cluster and simplify storage administration as they can share Oracle Automatic Storage Management disk groups. In larger clusters of more than two nodes, policy-managed databases can simplify management of many consolidated databases according to policies. Since consolidated databases (should) share a common database software installation, database consolidation creates no performance overhead.
11. **Prefer instance consolidation over zones when oversubscribing CPU**. Uncapped zones cannot make use of CPU resource management, which allows uncapped zones to compete for CPU with other zones without bounds.

With instance caging of consolidated database instances, a caged database instance can be capped in its maximum (foreground) CPU consumption.

12. **Use PDBs for high-density consolidation or DBaaS implementations**. PDBs have virtually no CPU or memory footprint, and can dynamically allocate both CPU and memory resources on demand. This allows oversubscription of CPU and memory and thus achieves a much higher consolidation density with potentially dozens of PDBs per core. Since resources are not statically partitioned, PDBs can dynamically react to changes in demand and do not require upfront dimensioning of resources. The ability to seamlessly migrate PDBs live between cluster nodes for load balancing and maintenance operations, and the easy cloning of PDBs for provisioning of new tenants, make PDB a good choice for DBaaS deployments.
13. **Deploy PDBs in large domains**. PDBs can achieve significant efficiency gains through elimination of duplicate functionality, which leads to higher aggregate performance on the same set of physical resources. The more PDBs are consolidated into a single CDB, the larger the efficiency gains compared to other deployments. A CDB in an LDom of one or multiple sockets in size will yield higher gains than a CDB in a small zone. Scale-up of a single CDB instance to many sockets may require tuning and potentially increases mutual performance impacts between PDBs. An instance crash of a large CDB also affects more tenants at once. If a single CDB instance becomes too large, consolidation of multiple CDBs in the same domain, or deployment in multiple smaller domains should be considered.
14. **Prefer PDBs over schema consolidation**. While schemas allow equally dense deployments as PDBs, they provide much weaker isolation. In particular, the lack of lifecycle and namespace isolation in schemas make PDBs the preferred deployment model. PDBs also provide cloning and live migration.
15. **Share storage servers and group them if appropriate**. Since it is not feasible to use dedicated storage servers per database unless only very few databases are consolidated, storage servers must be shared. However, if databases can be grouped into two categories, each group of databases could use a dedicated set of storage servers, and databases within a group share all storage servers of the group. This perfectly isolates storage servers between both groups of databases. If storage is strictly isolated, also database servers of both groups should be equally isolated and ideally be deployed on different physical servers or inside different PDoms.

Service Catalogs

In consolidated or cloud-based database-as-a-service environments, individual databases may have different requirements with respect to isolation, density, elasticity, efficiency, and availability. Rather than creating custom deployments for each of them, a standardization on a few deployment options, described through a *service catalog*, enables reduction of management and deployment complexity. A service catalog is a collection of documents and artifacts that describe the services an IT organization provides, and specifies how those services are delivered and managed. Typically, a service catalog defines multiple service levels, often labeled as *bronze, silver, gold, and platinum.* Each service level may specify certain performance, availability, or security guarantees.

Through combination of the deployment options described in this paper, IT organizations are able to create service catalogs according to their needs. The presented ratings and recommendations should help to chose the best deployment models for each of the desired service levels. The References section provides further information about service catalogs.

Deployment Example

The following example illustrates how various deployment options can be combined to deploy databases on two physical eight-socket servers. These servers could be SPARC T5-8 servers, or eight-socket PDoms of an Oracle SuperCluster M6-32 server. The example assumes a service catalog with five different service levels, resulting in five standardized deployment variants (A, B, C, D, and E) for a total of 57 databases (per server). To protect against server failures, databases can be clustered (using Oracle RAC) or replicated (using Oracle's Data Guard or Oracle GoldenGate) between the two servers.

A) Highest Isolation—a single database instance deployed in a two-socket LDom with two entire PCIe root complexes and dedicated storage servers

   This database does not share any hardware components (except the server) with other databases and is fully isolated in terms of performance, security, and faults (with the exception of server failures).

B) Strong Isolation—a single database instance deployed in eight-core LDoms with dedicated SR-IOV functions and shared storage servers

   Databases are strongly isolated and only share a common last-level CPU cache as well as the same storage servers. While they share I/O devices, they each own SR-IOV functions.

C) Density—four database instances consolidated in eight-core LDoms with dedicated SR-IOV functions and shared storage servers

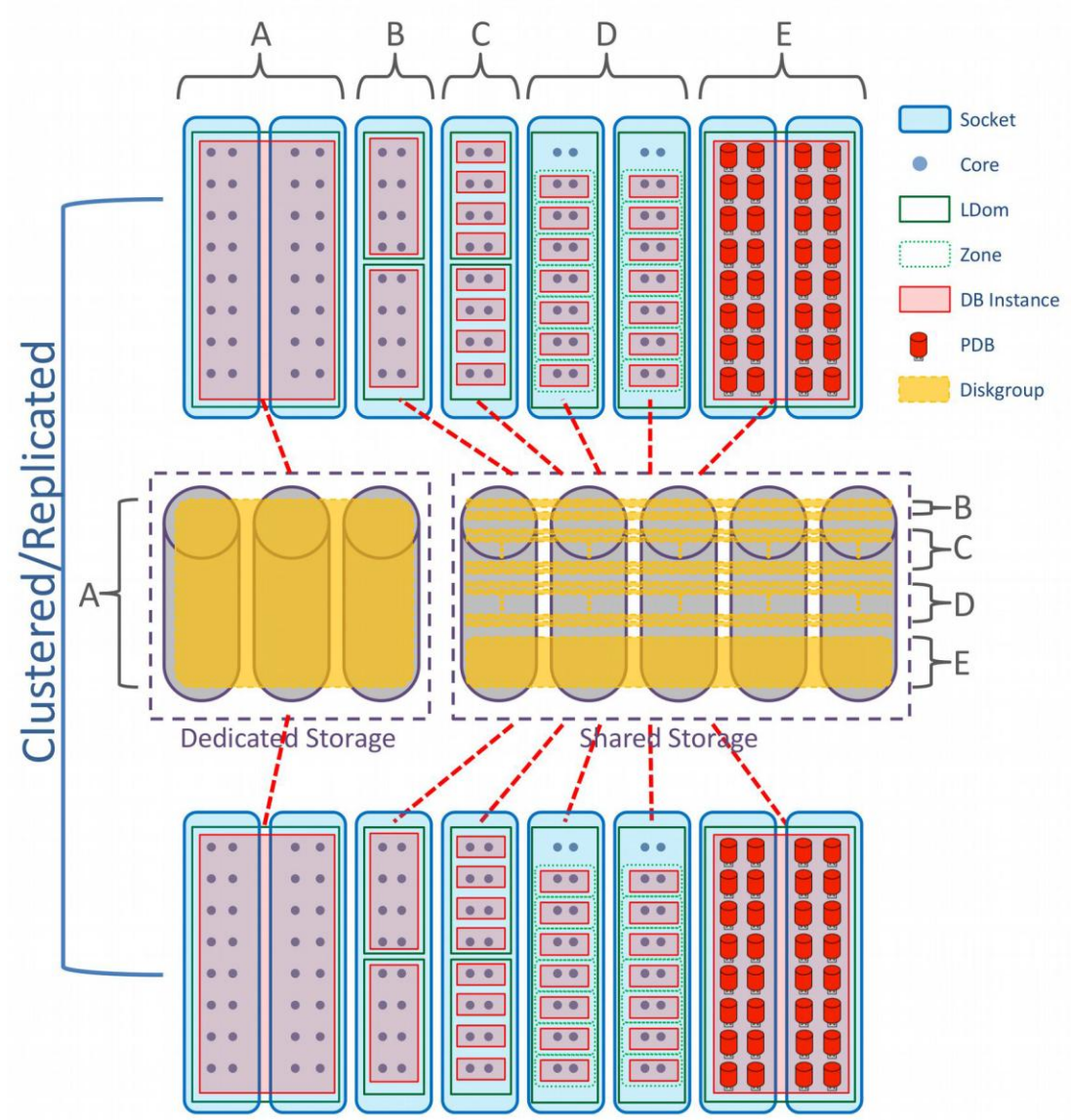

Figure 1: Deployment example

Eight databases are split into two groups of four databases each, which are isolated from each other (for example, for performance, availability, or security reasons). The four databases of one group share a common OS kernel and cluster (with its grid disks).

D) Density and Security—single database instances deployed in two-core zones inside single-socket LDoms with dedicated PCIe root complex and shared storage servers

Fourteen databases are each encapsulated in their own zone for security isolation between them. They each have their own cluster and grid disks. For improved locality of memory and fault isolation, the 14 databases are spread over two one-socket LDoms rather than being deployed in a single two-socket LDom.

E) High Density and Elasticity—32 PDBs inside a CDB deployed in a two-socket LDom with two entire PCIe root complexes and shared storage servers

   PDBs yield the highest savings from elimination of common functionality and resources if many of them are deployed in the same CDB. All 32 PDBs are therefore deployed in a fairly large two-socket LDom, in which they dynamically share resources on demand.

In the example, only databases are deployed on the two servers. SuperCluster also allows the deployment of applications on the same server, so some domains or zones also may be used to run applications.

Conclusion

Oracle SuperCluster offers different technologies operating at different layers that can be used to implement a consolidated database infrastructure. These technologies range from hardware partitioning over operating system virtualization to instance consolidation and in-database virtualization. Each consolidation option yields different results in terms of isolation, density, elasticity, efficiency, availability, and manageability. By combining these technologies, organizations can create consolidated database environments that perfectly fit their individual needs. This paper compares all consolidation options offered on Oracle SuperCluster with respect to isolation, density, elasticity, efficiency, availability, and manageability requirements and rates them in each of these categories. The evaluation and final recommendations should help to guide organizations in accomplishing the ideal balance of their requirements and designing a scalable, secure, efficient, and reliable infrastructure for database consolidation on Oracle SuperCluster.

References

1. Consolidation Using Oracle's SPARC Virtualization Technologies. Oracle Elite Engineering Exchange, September 2014. http://www.oracle.com/technetwork/server-storage/sun-sparc-enterprise/technologies/consolidate-sparc-virtualization-2301718.pdf
2. Oracle VM Server for SPARC Best Practices. An Oracle White Paper, June 2015. http://www.oracle.com/technetwork/server-storage/vm/ovmsparc-best-practices-2334546.pdf
3. Best Practices for Deploying Oracle Solaris Zones with Oracle Database 11*g* on SPARC SuperCluster. An Oracle Technical White Paper, November 2012. http://www.oracle.com/technetwork/server-storage/engineered-systems/sparc-supercluster/deploying-zones-11gr2-supercluster-1875864.pdf
4. Best Practices for Database Consolidation On Exadata Database Machine. An Oracle White Paper, October 2013. http://www.oracle.com/technetwork/database/features/availability/exadata- consolidation-522500.pdf
5. Oracle Multitenant. An Oracle White Paper, June 2013. http://www.oracle.com/technetwork/database/multitenant/overview/multitenant-wp-12c- 2078248.pdf
6. Oracle Multitenant on SuperCluster T5-8: Scalability Study. An Oracle White Paper, April 2014. http://www.oracle.com/technetwork/database/multitenant/learn-more/oraclemultitenantt5-8-final-2185108.pdf
7. Database Live Migration with Oracle Multitenant and the Oracle Universal Connection Pool on Oracle Real Application Clusters (Oracle RAC). An Oracle White Paper, October 2014. http://www.oracle.com/technetwork/database/multitenant/learn-more/pdblivemigration- 2301324.pdf
8. Oracle Real Application Clusters (Oracle RAC). An Oracle White Paper, June 2013. http://www.oracle.com/technetwork/database/options/clustering/rac-wp-12c-1896129.pdf
9. Service Catalogs: Defining Standardized Database Services. An Oracle White Paper, June 2014. http://www.oracle.com/technetwork/database/database-cloud/private/service-catalogs-for-dbaas-2041214.pdf

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

ORACLE

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Integrated Cloud Applications & Platform Services

Database Consolidation on Oracle SuperCluster
July 2015
Author: Nicolas Michael, Yixiao Shen