

**ORACLE WHITEPAPER**  
**APRIL 2016**

# HIGH AVAILABILITY BEST PRACTICES FOR DATABASE CONSOLIDATION

Oracle Maximum  
Availability Architecture

# High Availability Best Practices for Database Consolidation

The Foundation for Database-as-a-Service

ORACLE WHITE PAPER | APRIL 2016



## Table of Contents

Executive Overview	1
Introduction	2
Operating System Virtualization - Virtual Machines	2
Schema Consolidation	2
Database Consolidation with Oracle Database 11g	2
Database Consolidation Using Oracle Multitenant	3
Oracle Exadata - Optimized for Consolidation and DBaaS	5
Proof Points – High Consolidation Density Using Exadata	6
High Availability Reference Architectures	8
Database Consolidation Planning	10
Migration to the Multitenant Architecture	13
Recommended Migration Strategy	14
File Movement	15
Migration and Data Guard	15
Oracle Resource Management	18
Planning	18
Monitoring Resources	24
High Availability and Data Protection	26
Managing Unplanned Outages	28
Managing Planned Maintenance	30
Life Cycle Management for DBaaS	31
Conclusion	32
Appendix A: Exadata Consolidation Density and Performance	33
Appendix B: RMAN Active Database Duplication Migration	37



## Executive Overview

Enterprises are under intense pressure to do more with less, to reduce risk and increase agility. The aggressive consolidation of information technology (IT) infrastructure and deployment of Database as a Service (DBaaS) on public or private clouds is a strategy that many enterprises are pursuing to accomplish these objectives.

Several key elements are needed to realize the full potential for cost reduction through database consolidation and DBaaS. High consolidation density and management simplicity are required to achieve maximum reduction in hardware and administrative costs. These attributes must then be combined with intelligent software infrastructure capable of achieving service level agreements (SLAs) for availability, performance, and data protection.

Oracle Multitenant with Oracle Database 12c represents a fundamental re-architecting of the Oracle Database to make it the best platform for database consolidation. Oracle Multitenant is evolutionary, making it simple to consolidate existing Oracle databases to reduce cost. Oracle Multitenant is also revolutionary, enabling maximum consolidation density and dramatically simplifying the management of consolidated database environments compared to previous consolidation strategies.

Oracle Multitenant with Oracle Maximum Availability Architecture (MAA) provides high availability and data protection for consolidated database environments where the impact of any outage would otherwise be magnified many times over. While Oracle Multitenant and MAA provide unique benefit regardless of the underlying hardware platform or operating system environment, they provide maximum benefit when deployed on Oracle Engineered Systems. Oracle integrated hardware and software solutions reduce total life cycle costs by using standard, high performance platforms that achieve economies of scale for consolidated environments along multiple dimensions: performance, reliability, manageability, and support. Oracle Exadata Database Machine, for example, has demonstrated up to 5x advantage in consolidation density compared to traditional systems.

This paper provides MAA best practices for Database Consolidation using Oracle Multitenant. It describes standard HA architectures that are the foundation for DBaaS. It is most appropriate for a technical audience: Architects, Directors of IT and Database Administrators responsible for the consolidation and migration of traditional database deployments to DBaaS. Recommended best practices are equally relevant to any platform supported by Oracle Database except where explicitly noted as being an optimization or an example that applies only to Oracle Engineered systems.

## Introduction

This paper begins with a discussion of traditional approaches to database consolidation and their trade-offs that provides context for the high value provided by Oracle Multitenant. The remainder of the paper presents HA best practices for database consolidation and includes service-level driven standard reference architectures for high availability and data protection: Bronze, Silver, Gold and Platinum.

### Operating System Virtualization - Virtual Machines

Early consolidation efforts focused on server consolidation using operating system virtualization. Operating system virtualization offers simple, template-driven deployment and the ability to increase server utilization by carving up a single physical machine into multiple virtual machines (VMs). Increasing server utilization is one of the goals of database consolidation. Rather than run ten databases on ten separate physical machines each with varying degrees of utilization, server footprint can be reduced by deploying these same databases in ten virtual machines (VMs) on a single physical system.

From the perspective of database consolidation, however, there are several shortcomings to what VMs can achieve on their own:


- » Consolidation density is limited due to the inefficiency of running multiple instances of operating system and database software each with their accompanying memory and systems footprint. This causes performance to suffer as more VMs and databases are deployed on a given physical machine.
- » VMs are limited in their ability to control administrative costs because they do nothing to reduce the total number of database and operating system environments that must be managed.
- » VMs do not address the HA requirements of mission critical applications that cannot tolerate downtime or data loss. VMs are limited to cold restart following an outage and they depend upon integration with external replication technologies to provide real-time data protection.
- » VMs are limited in providing dynamic resource management for Oracle Database where consumption of system resources (e.g. CPU or IO) may vary throughout the day from one consumer group or database to another in a database consolidated environment due to business cycles.

### Schema Consolidation

Schema consolidation describes the process of transforming standalone databases into schemas that are consolidated into a single database. This addresses the shortcomings in consolidation density and management efficiency inherent in using VMs as a consolidation strategy for the database tier. While there are many cases of successful deployments, schema consolidation has its own set of trade-offs. It is challenging to implement, particularly with existing applications and database environments that were not designed for schema consolidation. Schema consolidation also does not possess the inherent simplicity of portability and isolation at a schema level compared to what can be achieved with a VM.

### Database Consolidation with Oracle Database 11g

Oracle Database using Oracle Real Application Clusters (Oracle RAC) and Oracle Resource Management provided the first consolidation platform optimized for Oracle Database and is the MAA best practice for Oracle Database 11g. Oracle RAC enables multiple Oracle databases to be easily consolidated onto a single Oracle RAC cluster. Oracle Resource Management enables the prioritization of system resources across multiple databases and consumer groups consolidated on an Oracle RAC cluster to ensure that service level objectives for response time are achieved.



Oracle Exadata Database Machine provides additional value for consolidated environments through the combination of scalable performance and unique resource management capabilities. For more details see [“Best Practices for Database Consolidation on Oracle Exadata Database Machine.”](#)

Oracle RAC, Exadata and Resource Management offer a simple method of consolidation with substantial manageability, performance, and HA benefits, but they are unable to achieve the same efficiency as schema consolidation. Each database (system and user data that resides on disk) in an Oracle RAC cluster has one or more database instances (dedicated background processes and memory area). Each database instance can mount and open a single database. As more databases are consolidated in a single cluster, the amount of memory and CPU that must be dedicated to each database instance presents a practical limit to the consolidation density that can be achieved.

### Database Consolidation Using Oracle Multitenant

Oracle Multitenant fundamentally changes Oracle Database architecture by introducing the concepts of multitenant container databases (CDB) and pluggable databases (PDB). Existing databases can be easily converted to a PDB. Consolidation is achieved by ‘plugging in’ multiple PDBs into a single CDB<sup>1</sup>. Oracle Database 12c with Oracle Multitenant is engineered to deliver the most efficient platform in every aspect for database consolidation.

A CDB has a single set of background processes and shared memory area (SGA) that is used by all PDBs. This architecture requires less CPU and memory compared to traditional approaches of consolidating multiple independent databases onto a single physical machine, multiple VMs, or an Oracle RAC cluster. While a CDB can be deployed in either physical or virtual environments, it achieves the highest management and performance efficiency for the database tier when deployed on a physical machine. The CDB itself becomes the virtualization technology for the database tier, eliminating the overhead of multiple VMs and guest operating systems.

The advantages of the Oracle Multitenant Architecture were demonstrated by a series of tests performed on a Sun T5-8 server running Oracle Database 12c. The T5-8 was configured with 128 cores, 2 TB of memory and 8 Exadata Storage Servers. Tests were run using 252 consolidated OLTP databases (33% small at 1 GB, 33% medium at 5 GB and 33% large at 15 GB). Tests were designed to compare deployment of pluggable databases in a multitenant architecture to single instance databases.

Testing proved that Oracle Multitenant can increase consolidation density compared to single-instance databases using the same system resources (CPU, memory, and I/O). Multitenant achieved:

- » A 50% increase in consolidation density (the number of databases consolidated) while achieving the same throughput per database
- » An 80% increase in aggregate throughput when consolidating the same number of databases

When viewed from another perspective the tests demonstrated how Oracle Multitenant produces real savings in hardware and software licensing costs:

- » A multitenant architecture required 64 fewer CPU cores and 1/3 the IOPS (I/O per second) to consolidate an equivalent number of databases (252), with the same level of throughput

Figure 1 provides a summary of test results. For additional details refer to the [Oracle Multitenant consolidation study](#) published on the Oracle Technology Network.

---

<sup>1</sup> In Oracle Database 12c Release 1 (12.1), a CDB can contain up to 252 PDBs. An increase in this limit is planned in future releases.

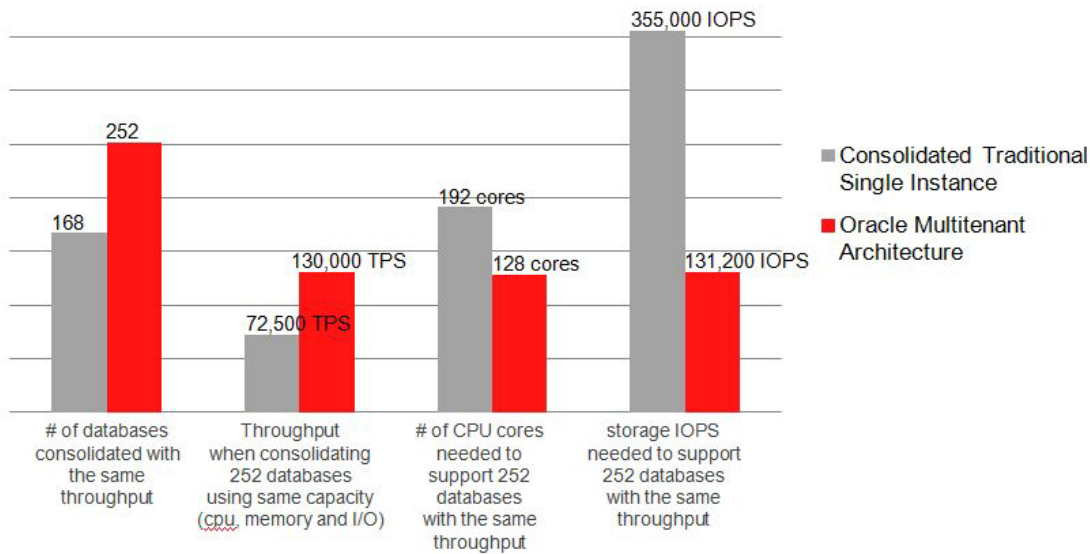


Figure 1: Consolidation Advantages of Oracle Multitenant Architecture

Oracle Multitenant also offers the simplicity and efficiency to manage many-as-one. A simple illustration of this is provided in Figure 2. Regardless of how many PDBs are consolidated in a CDB there is still only one database to manage (the multitenant container database): a single backup, a single copy for disaster recovery (DR), and a single database to upgrade and patch. Applying this principle to the relatively small consolidated environment represented in Figure 2, Oracle Multitenant achieves a 4 to 1 advantage by reducing the number of separate databases that require management. This benefit continues to grow as more databases are consolidated within a single CDB.

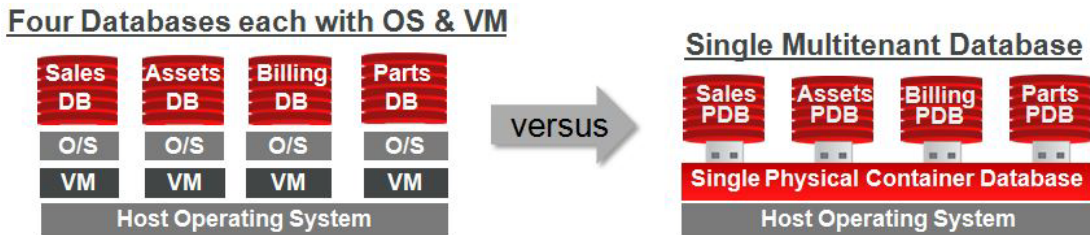


Figure 2: Consolidation using Virtual Machines compared to Database Virtualization using Oracle Multitenant

Oracle Multitenant also provides a high degree of isolation. A PDB can be easily unplugged from one CDB and plugged into another to allow database administrators the option of performing maintenance on an individual PDB if required. An individual PDB can be provisioned, patched, cloned, consolidated, restored, or moved without impacting other PDBs in the same CDB.

Oracle Multitenant is unique in accomplishing the positive attributes of alternative consolidation methods while avoiding each of their drawbacks. Oracle Multitenant achieves:

- » The simplicity and flexibility of VMs, without the limits to consolidation density, performance, or increased management complexity
- » The high consolidation density of schema consolidation, without the implementation complexity, limited flexibility and limited isolation

- » The HA, scalability, and automated workload management of simple database consolidation using Oracle RAC with Oracle Database 12c, without the limitations in consolidation density or management complexity of a separate database (each with its own operational overhead) for each application

Oracle Multitenant seamlessly integrates with the HA and data protection capabilities of Oracle Database. This integration combined with Oracle Maximum Availability Architecture (MAA) best practices provides an evolutionary upgrade path to a revolutionary technology for database consolidation.

## Oracle Exadata - Optimized for Consolidation and DBaaS

[Oracle Engineered Systems](#) are a family of systems that implement various optimizations for the Oracle Database. They include: Oracle Exadata Database Machine, Oracle SuperCluster, Oracle Database Appliance, and Oracle Virtual Compute Appliance. Oracle Engineered systems reduce lifecycle cost by standardizing on a pre-integrated and optimized platform for Oracle Database, hardware, and software supported by Oracle.

[Oracle Exadata Database Machine](#) is the engineered system purpose-built to provide optimal performance and manageability for Oracle Database. Its scalable architecture and advanced software capabilities make it ideally suited as a standard database platform for consolidation and DBaaS.

The unique capabilities of Exadata Storage software include:

- » Database processing offload in storage

Queries run significantly faster by pushing database expression evaluations to the storage cells for processing. Only the rows satisfying the predicate, specified columns, and predicated columns are returned to the database server, eliminating unproductive data transfer to the database server. This can result in 100 GB per second SQL data throughput with a fully configured Oracle Exadata Database Machine.

- » Database optimized storage

Hybrid columnar compression compresses read-intensive and archival data to 10 times or more, reducing your overall storage footprint requirements.

- » Database optimized PCI flash

Exadata Smart Flash Cache transparently caches frequently accessed data to fast solid-state storage, improving query response times and throughput. Write operations serviced by flash instead of by disk are referred to as “write back flash cache.” The caching is intelligent in that backups and infrequent table scans are not cached. Starting with Exadata 11.2.3.3, Oracle Exadata Storage Server Software automatically caches objects read by table and partition scan workloads in flash cache based on how frequently the objects are read without impacting Online Transactional Processing (OLTP) or over burdening the flash cache, while simultaneously compressing the objects. PCI flash eliminates disk controller bottlenecks.

Exadata Smart Flash Log provides consistent low-latency log writes using only 512 MB of space on each flash disk. The consistent low-latency log writes can stabilize and improve OLTP workloads even in mixed workloads.

Exadata Smart Flash Cache Compression (with Exadata 11.2.3.3 or later) dynamically increases the logical capacity of the flash cache by transparently compressing user data as it is loaded into the flash cache. This allows much more data to be kept in flash and decreases the need to access data on disk drives. The I/Os to data in flash are orders of magnitude faster than the I/Os to data on disk. The compression and decompression operations are completely transparent to the application and database, and they have no performance overhead even when running at rates of millions of I/Os per second.

- » Database optimized and comprehensive resource management

Oracle Database resource management is available on all platforms supported by Oracle Database. Examples of capabilities include guaranteeing CPU to PDBs and database workloads based on their priority, managing and pacing parallel operations, and detecting and controlling runaway queries, as described in [Managing Resources with Oracle Database Resource Manager](#). Exadata, however, enables additional unique resource management capabilities that are essential for supporting DBaaS. These include I/O Resource Management (IORM) and Network Resource Management.



IORM enables multiple databases, PDBs, and database workloads to share the same storage while ensuring they utilize I/O resources according to their priority. Oracle Exadata Storage Server Software works with IORM and Oracle Database Resource Manager to ensure that customer-defined policies are met, even when multiple databases, CDBs, or PDBs share the same database cluster and storage grid. As a result, one database or application service cannot monopolize the I/O bandwidth and degrade the performance of the other databases in an unpredictable way. Starting with Exadata 11.2.3.2.1, IORM supports up to 1024 databases on Exadata. Starting with Exadata 12.1.1.1, IORM has been extended to support individual PDBs within the multitenant architecture in Oracle Database 12c.

Starting in Exadata 11.2.3.3 and 12.1.1.1 or later, Network Resource Management automatically and transparently prioritizes critical database network messages, ensuring fast response times for latency-critical operations. Prioritization is implemented in the database node, database InfiniBand adapters, Oracle Exadata Storage Server Software, Exadata storage cell InfiniBand adapters, and InfiniBand switches to ensure prioritization happens through the entire InfiniBand fabric. Latency-sensitive messages, such as Oracle RAC Cache Fusion messages and critical background inter-node communication, are prioritized over batch, reporting, and backup messages. Log file write operations are given the highest priority to ensure low latency for transaction processing even in the presence of large batch loads, backups, recover reads and writes, and intensive reports and queries that can saturate the network.

Exadata's advanced features and unique quality of service capabilities make it the only platform that provides end-to-end prioritization from application to database to network to storage to intelligently manage workloads in database consolidated environments and for DBaaS.

## Proof Points – High Consolidation Density Using Exadata

Oracle conducted tests to validate Exadata's ability to achieve substantially higher consolidation densities than a similarly configured non-Exadata system. Two series of tests were run on an Oracle Exadata Database Machine using an identical workload and Oracle Database configuration.

- » The first series of tests disabled all Exadata-specific features to provide a baseline for comparison. While the intention was to simulate a non-Exadata system, this approach provided a conservative baseline for comparison given the high I/O and network bandwidth and robust performance characteristics of an Exadata system even without Exadata features enabled. The performance of this 'non-Exadata' system far exceeds that of generic Linux systems.
- » The second series of tests ran the same workload as the first and on the same machine, but this time with all Exadata-only features enabled, including Exadata Smart Flash Logging, Smart Flash Cache, Smart Scan, Smart Flash Cache Compression, Storage Indexes, Network Resource Manager, and I/O Resource Manager. This made it possible to measure the ability of Exadata's unique features to increase consolidation density for database workloads.
- » Oracle Multitenant was not used in either series of tests in order to isolate the value of Exadata's unique capabilities for database consolidation.

The summary provided in Figure 3 illustrates results that answer two basic questions:

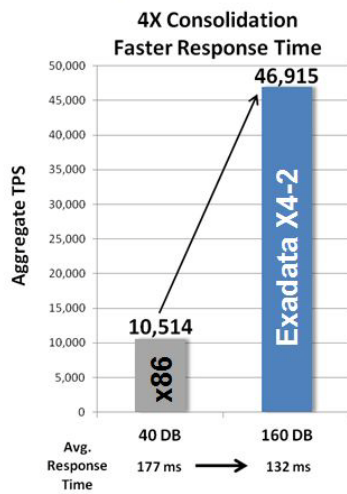
- » How many more databases can be consolidated on an Exadata system compared to a similarly configured non-Exadata x86 system?

The first test simulated OLTP workload where the number of databases being consolidated was increased until a bottleneck was reached. Exadata provided faster response times while demonstrating 4 times greater consolidation density by supporting 160 databases (CPU bound) compared to just 40 databases (I/O bound) on equivalent X86 hardware. In consolidated environments where performance service level agreements allow for oversubscription, the consolidation density advantage of an Exadata system grew to 5 times that of a similarly configured x86 system.

- » How much faster can Exadata execute mixed workloads that are typical of consolidated environments compared to a similarly configured x86 system?

The workload simulated in this test included OLTP workload and a reporting data warehouse. Tests showed that Exadata had the advantage of 15 times greater transaction response times combined with 2 times greater consolidation density and more than 6 times greater transaction volume. An additional interesting aspect of this test is that the non-Exadata system was I/O bottlenecked at 40 databases while the Exadata system had additional I/O and CPU capacity to address spikes in workload or to consolidate additional databases.

How many more databases can be consolidated on Exadata?



How much faster can Exadata execute mixed workloads?

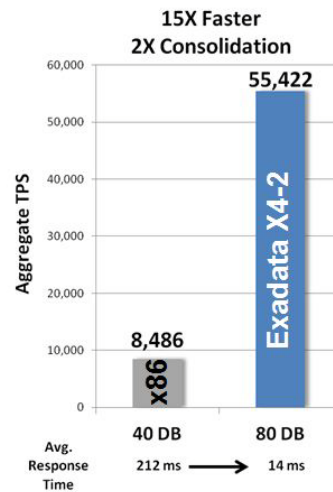


Figure 3: Exadata Consolidation Density and Performance

The increase in consolidation density enables Exadata to generate substantial cost benefits: less hardware to purchase, fewer systems to manage, reduced power consumption, and fewer Oracle Database licenses required.

Details of the methodology, configuration, and results for the tests described in this section and others are provided in [Appendix A](#).

## High Availability Reference Architectures

Oracle MAA best practices define four HA reference architectures that address the complete range of availability and data protection required by enterprises of all sizes and lines of business. The architectures, or HA tiers, are designated PLATINUM, GOLD, SILVER, and BRONZE. They deliver the service levels described in Figure 4.

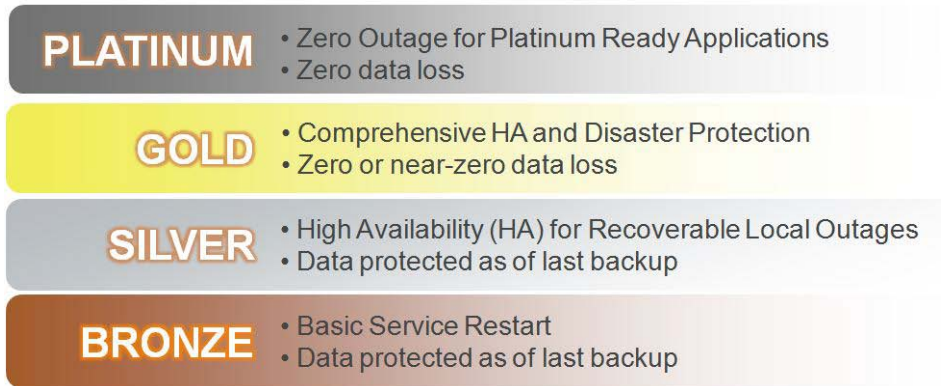


Figure 4: HA and Data Protection Service Levels

Each tier uses a different MAA reference architecture to deploy the optimal set of Oracle HA capabilities that reliably achieve a given service level at the lowest cost and complexity. They explicitly address all types of unplanned outages including data corruption, component failure, system and site outages, as well as planned outages due to maintenance, migrations, or other purposes. A high-level description of each architecture is provided Figure 5.

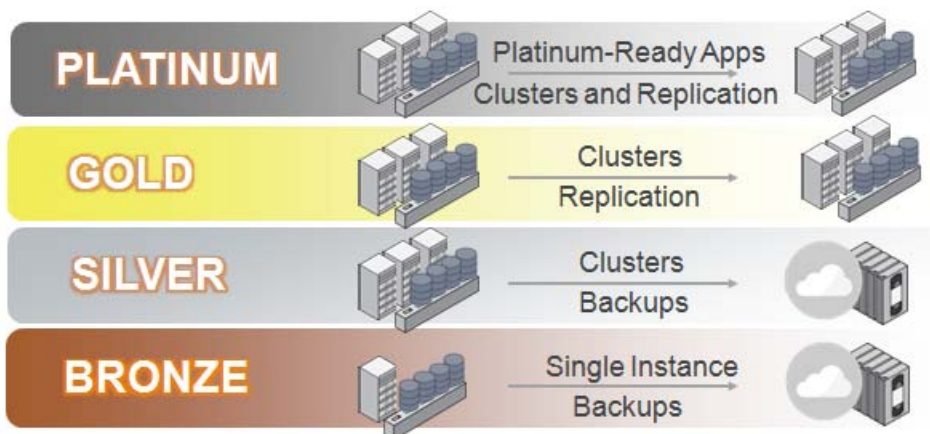


Figure 5: HA and Data Protection Reference Architectures

**Bronze** is appropriate for databases where simple restart or restore from backup is 'HA enough'. Bronze is based upon a single instance Oracle Database using MAA best practices that incorporate the many data protection and HA capabilities included with an Oracle Enterprise Edition license. Oracle-optimized backups using Oracle Recovery

Manager (RMAN) provide data protection and are used to restore availability should an outage prevent the database from being able to restart.

**Silver** provides an additional level of HA for databases that require minimal or zero downtime in the event of database instance or server failure as well as many types of planned maintenance. Silver adds clustering technology - either Oracle RAC or RAC One Node. RMAN provides database-optimized backups to protect data and restore availability should an outage prevent the cluster from being able to restart.

**Gold** substantially raises the service level for business critical applications that cannot accept vulnerability to single points-of-failure. Gold adds database-aware replication technologies, Active Data Guard and Oracle GoldenGate, which synchronize one or more replicas of the production database to provide real time data protection and availability. Database-aware replication greatly increases HA and data protection beyond what is possible with storage replication technologies. It also reduces cost while improving return on investment by actively utilizing all replicas at all times.


**Platinum** introduces several new Oracle Database 12c capabilities as well as previously available products that have been enhanced with the latest release. These include Application Continuity for reliable replay of in-flight transactions that masks outages from users; Active Data Guard Far Sync for zero data loss protection at any distance; new GoldenGate enhancements for zero downtime upgrades and migrations; and Global Data Services for automated service management and workload balancing in replicated database environments. While each technology requires additional effort to implement, they deliver substantial value for the most critical applications where downtime and data loss are not an option.

The HA and data protection attributes inherent to each reference architecture are summarized in Table 1.

TABLE 1: HIGH AVAILABILITY AND DATA PROTECTION

Outage class/ HA tier	Unplanned Outages (local site)	Planned Maintenance	Data Protection	Unrecoverable local outages and disaster recovery
Platinum	Zero outage for platinum ready applications	Zero application outage	Comprehensive runtime validation combined with manual checks	Zero application outage for platinum-ready applications, in-flight transactions are preserved, zero data loss
Gold	Comprehensive HA/DR	All rolling or online	Comprehensive runtime validation combined with manual checks	Real-time failover, zero or near-zero data loss
Silver	HA with automatic failover	Some rolling, some online, and some offline	Basic runtime validation combined with manual checks	Restore from backup, potential to lose data generated since last backup
Bronze	Single instance with auto restart for recoverable instance and server failures	Some online, most offline	Basic runtime validation combined with manual checks	Restore from backup, potential to lose data generated since last backup

The MAA reference architectures provide a standard infrastructure optimized for Oracle Database that enables enterprises to dial-in the level of HA appropriate for different service level requirements. Standardization reduces



cost and makes it simple to move a database from one HA tier to the next, or from one hardware platform to another should business requirements change.

Refer to the MAA best practice paper [Oracle MAA Reference Architectures](#) for additional details of Oracle capabilities and the service levels achieved by each of the reference architectures.

## Database Consolidation Planning

The first step in planning HA for consolidated databases begins at the same place as for any database. A business impact analysis is performed to assess the tolerance for data loss (recovery point objective, or RPO) and downtime (recovery time objective, or RTO) for each database that is a candidate for consolidation. This analysis also identifies any dependencies that may exist between two or more databases where unavailability or data loss for one database would impact the ability of other databases to effectively service the applications they support.

### Group Candidates for Consolidation into HA Tiers

The second step is to identify sets of databases that are eligible for consolidation with each other based upon their RTO and RPO. The MAA best practice is to consolidate databases that have similar RTO and RPO according to the standard set of HA tiers described in the preceding section. Note that in the case where there are dependencies between databases, each database is assigned to the HA tier appropriate for the database having the most stringent HA requirement.

The process of grouping databases into standard HA tiers according to RTO and RPO requirements accomplishes three objectives:

- » Standardization on a limited set of HA tiers reduces complexity and achieves economies of scale.
- » Efficient consolidation by avoiding over-investment or unnecessary complexity in HA infrastructure and processes. For example, it would be inefficient to consolidate a database with an RTO and RPO that can be achieved by restoring a backup with another database that has an RTO and RPO that requires additional HA infrastructure for real-time HA and DR.
- » Establishing the HA and data protection component of the [Service Catalog for DBaaS](#). The service catalog describes the services that an IT organization provides its user community - developers, architects, and end-users - with specifics on how database services are delivered and managed.

### Select a Consolidation Method

The reference architecture for each HA tier supports all consolidation methods and deployment models. However, detailed best practices for the migration of standalone databases to a consolidated environment are influenced by the consolidation method used.

- » Server consolidation using a VM deployment model is similar from a database perspective to the migration of multiple independent Oracle Databases from disparate machines onto a single physical machine. In this regard the HA best practices included in this paper apply equally to physical or virtual environments. However, VMs add considerations for capacity planning, performance, system management, and HA that create an additional level of complexity. These considerations are specific to the vendor of the virtualization technology used. Detailed best practices for providing HA for VMs, or for using VMs as a consolidation method, are outside the scope of this paper.
- » Schema consolidation is challenging to implement for existing applications. Challenges include name collisions, security concerns, difficulty with per-schema point in time recovery, patching, and cloning. Best practices for migrating standalone databases to a schema-consolidated environment are outside the scope of this paper.

- » Simple database consolidation using Oracle RAC combined with Oracle Resource Manager is the MAA best practice for Oracle Database 11g. Except where Oracle Database 12c features are explicitly noted, the HA tiers documented in this paper are equally relevant to both Oracle Database 11g and Oracle Database 12c.
- » Exadata consolidation best practices for Oracle Database 11g are outside the scope of this paper, but they are described in detail in [“Best Practices for Database Consolidation on Oracle Exadata Database Machine.”](#)
- » Oracle Multitenant is the optimal database consolidation method from Oracle Database 12c onward. The multitenant architecture combines the best attributes of each of the previous consolidation methods without their accompanying tradeoffs.

This paper assumes that Oracle Multitenant is the database consolidation method utilized beginning with Oracle Database 12c onward. MAA best practices are equally relevant to physical or virtual environments, but they do not take into account additional considerations that are unique to VMs.

### Align HA Tiers with Hardware Pools

At a high level, the process of virtualizing an Oracle Database and consolidation using Oracle Multitenant is as simple as converting each database into a PDB and consolidating them into one or more CDBs for a given HA tier. Ultimately you must perform the analysis necessary to provision enough capacity to deliver the required service levels for the databases to be consolidated in each tier. Part of the capacity planning exercise requires projecting your ultimate consolidation density in terms of how many CDBs will be deployed in each tier. There are several reasons to deploy multiple CDBs within an HA tier even when all PDBs in a given tier have a similar service level requirement.

- » The underlying hardware may have a capacity limit that prevents all PDBs assigned to a particular HA tier from being consolidated into a single server or cluster of servers. Performance requirements must be assessed to determine the suitability of a database as a candidate for database consolidation, and to match existing managed servers that you want to consolidate with the servers that you will consolidate to.
- » It is often necessary to support more than one Oracle Database release/patch set at a time. For example, one CDB may be at Oracle Database 12c Release 1 (12.1.0.1) and a second CDB at Oracle Database 12.1.0.2. This enables an individual PDB to be ‘unplugged’ from an Oracle 12.1.0.1 CDB and ‘plugged’ into an Oracle 12.1.0.2 CDB to implement an upgrade should other 12.1 PDBs be unable to upgrade. This can happen, for example, due to a delay in a vendor-supplied application supporting the new release.
- » You may be conservative and prefer to deploy CDBs with a more limited number of PDBs to gain experience with the multitenant architecture. There is no future penalty for this strategy given that it is easy to increase consolidation density simply by unplugging a PDB from one CDB and plugging it into another, eventually reducing the number of CDBs in each HA tier.

There is also the classic question of which databases should be the first to migrate to Oracle Multitenant. The Bronze tier will have the largest number of databases eligible for consolidation. Oracle recommends starting with the Bronze tier with the assumption that it will yield fast return on investment with little risk.

An overview of the consolidation planning process is described in Table 2.

TABLE 2: CONSOLIDATION PLANNING

Key Steps for Consolidation	Recommendations	Benefits
1. Standardize HA SLAs and assign applications and their databases to the appropriate HA Tier	Follow MAA HA Tiers (PLATINUM, GOLD, SILVER, BRONZE)	Reduce operational expense (OPEX) and lowers risk through standardization.
2. Design HA Architecture to meet SLAs, determine configuration and operational best practices	Follow the recommended HA reference architectures each HA Tier Use MAA configuration and operational best practices for Oracle Database ( <a href="http://www.oracle.com/goto/maa">www.oracle.com/goto/maa</a> )	Reduce OPEX Increase overall stability and availability. MAA validated configuration best practices reduce risks.
3. Reduce the number of hardware platforms and operating systems for the Oracle Database platform	For Linux or Solaris x86-64, Exadata Database Machine provides a standard, scalable database platform to achieve maximum consolidation density.  Oracle Database Appliance provides a standard platform with high consolidation density for smaller environments  For SPARC and Solaris or multi-purpose database and application processing, use Oracle SuperCluster (includes Exadata storage and Exadata features)  Oracle Virtual Compute Appliance optimized for deploying virtual infrastructures for any Linux, Oracle Solaris, or Microsoft Windows application	MAA best practices are optimized and preconfigured for engineered system platforms Reduce setup time and OPEX Integrated support of both systems and software reduces OPEX Reduce CAPEX since fewer hardware platforms. Exadata software combined with Oracle resource management and compression technologies are optimized for database consolidation High consolidation density reduces total life-cycle cost.
4. Establish hardware pools that are targets for database consolidation	Create appropriately sized hardware pools. Recommend Half Racks up to two full inter-rack systems for a typical hardware pool.  Assign each hardware pool to an HA Tier. Since each tier has a different HA architecture, there is only one HA tier per hardware pool.	Reduce complexity that accompanies system sprawl and variability Reduce OPEX
5. Choose standard image software stack	Choose 2 or 3 latest Oracle patchsets after internal validation. Examples include 11.2.0.3, 11.2.0.4 and 12.1.0.1 with the recommended PSU or Bundle Patches.  Choose no more than 1 other variant per database patchset release. One Oracle Grid Infrastructure and maximum five Oracle Homes are recommended.	Reduce software sprawl and risk. Reduce OPEX and increase overall stability and availability.
6. Choose database consolidation method	Use multitenant architecture in Oracle Database 12c for the most efficient form of database consolidation.  Use operating system virtualization such as OVM and Oracle Solaris Zones if dedicated resources are required for strict isolation and the resulting increase in OPEX and CAPEX costs are acceptable.	For the greatest savings in OPEX and capital expenditures
7. System sizing, resource requirements, and performance expectations	Evaluate current and future CPU, I/O, memory, and storage capacity consumption for each application.  Use EMCC Consolidation Planner, (see <a href="#">documentation</a> and <a href="#">demonstration</a> ) or contact Oracle Consulting or Oracle Advanced Customer Support Services (ACS) for sizing services.	Reduce CAPEX and OPEX after migration  Data will be used to assign databases to Hardware Pools and to configure Resource Manager allocations and limits.

8. Assign databases to appropriate hardware pools with the same HA Tier	<p>If consolidating into a Hardware Pool that already hosts active databases, monitor the current CPU, I/O, memory, and storage consumption to understand its available capacity.</p> <p>When a Hardware Pool has no spare capacity, assign the next database to a new Hardware Pool. There will be cases where there is a single database in a hardware pool because of its criticality and resource consumption.</p> <p>Choose EM12c for provisioning</p>	
9. Configure resource allocations and limits	<p>Use Resource Management best practices to guarantee and limit resources for each database and PDB.</p> <p>For databases that require dedicated resources for isolation, consider using the <code>PROCESSOR_GROUP_NAME</code> parameter to bind the instance to dedicated CPUs or NUMA nodes.</p> <p>Refer to the Resource Management section of this paper for more details.</p>	Efficient resource utilization
10. Deploy monitoring and self-service infrastructure	Oracle Enterprise Manager Cloud Control 12c	Reduce OPEX, improve service

## Migration to the Multitenant Architecture

Once you have completed consolidation planning and assigned candidates for database consolidation to approximate hardware pools and HA tiers, the next step is to migrate existing databases to the multitenant architecture. The method used is dependent upon:

- » The source database type (non-CDB, CDB, or PDB)
- » Source database version
- » Source and target hardware platform and database HA requirements

This section provides strategies to migrate your databases to the multitenant architecture. For generic migration options, refer to [Database Platform or Location Migration](#), or for Exadata migration best practices to non-multitenant databases refer to [Best Practices for Migrating to Exadata Database Machine](#).

With the release of Oracle Database 12c, the three database types are the following:

- » Non-container database (non-CDB). This is any database that is not a multitenant container database (CDB) or a pluggable database (PDB). It will include all pre-Oracle 12c release databases, any pre-Oracle 12c database that has been upgraded to Oracle 12c and not migrated to a PDB, and any Oracle 12c created database that was not created as either a CDB or a PDB.
- » Multitenant container database (CDB). A CDB is a database created in Oracle Database 12c with the `ENABLE_PLUGGABLE_DATABASE` initialization parameter set to `TRUE`. It is created to be a container of 0, 1, or many PDBs. It is not possible to convert a non-CDB into a CDB or vice versa since a CDB cannot be used to create schema objects or process user SQL. Oracle recommends that all new Oracle Database 12c databases be created as CDBs with application data stored in PDBs.
- » Pluggable database (PDB). This is a collection of portable schemas, schema objects, and non-schema objects that appear to an application as a separate database. Non-CDBs can be converted into a PDB, but a PDB cannot be converted into a non-CDB.



## Recommended Migration Strategy

Migration options to the multitenant architecture are highly dependent on the source database version and the desired HA tier discussed earlier in this paper. Table 3 summarizes the migration options. Also refer to [My Oracle Support Note 1576755.1: Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage](#).


TABLE 3: METHODS FOR MIGRATION TO A MULTITENANT ARCHITECTURE

HA Tier / Source	Migration Method	Considerations	Data Guard Standby Impact	Downtime Estimate
SILVER, GOLD or PLATINUM Any DB type Any DB version Any platform	GoldenGate	Requires a PDB to be precreated from seed prior to GoldenGate instantiation If using Data Pump or any logical instantiation, objects can be reorganized and re-optimized (for example, for compression). Fast fallback option available	Instantiation of Oracle GoldenGate objects on primary may require additional steps to maintain standby CDB	Near Zero based on manual switchover operation
SILVER Oracle Database 12c non-CDB AND Same-endian platform	RMAN Active Database Duplication HA/DR	Performing describe/non-CDB plugin Incremental file migration supported Requires no staging area This is a physical migration so objects are not reorganized or re-optimized (for example, for compression).	Copy files to standby CDB MRP continues on as it finds the files and MRP continues	45min, impacted by time for last incremental and time required for noncdb_to_pdb.sql execution
BRONZE or SILVER Any Oracle Database 11.2.0.3 or higher Any Platform	Full Database TTS/Data Pump	Requires a PDB to be pre-created from seed prior to Data Pump import Incremental file migration supported Requires staging area for incremental backups This is a physical migration so objects are not reorganized or re-optimized (for example, for compression).	Copy files to standby using ASMCMD cp or host based copy command Follow Data Guard documentation for handling the data files	2 hours, impacted by time required for Data Pump metadata import
BRONZE Any Oracle Database release Any platform	Transportable Tablespace (TTS) and DataPump export/import Custom scripts to copy or extract and load data.	Requires a seed PDB to be pre-created prior to Data Pump import If using Data Pump or any logical instantiation, objects can be reorganized and re-optimized (e.g for compression).	N/A	Dependent on the size and time to export and import Hours to days.

While it is possible to migrate directly from an Oracle Database release previous to Oracle Database 12c into a PDB using Transportable Tablespace and Data Pump, the preferred method is to first upgrade your database to Oracle Database 12c. The migration advantages of converting to the multitenant architecture from an Oracle Database 12c source include:

- » Simplicity. Fewer steps and the availability of more tools to simplify the process
- » Validating the performance and function of your Oracle Database 12c database prior to migration to the multitenant architecture

If you choose to upgrade your database first, note that Oracle Database versions 10.2.0.5, 11.1.0.7, and Oracle 11g Release 2 versions 11.2.0.2 or later can upgrade directly to Oracle Database 12c. All other versions will require two



upgrades by first upgrading the database to one of the previously listed minimum interim releases. Please refer to My Oracle Support Note 1462240.1: Oracle Database 12c Upgrade Companion, for more information about upgrading from a previous release to Oracle Database 12c on any platform. For Exadata systems please refer to My Oracle Support Note 1681467.1 to upgrade Grid Infrastructure and Oracle Database to 12.1.0.2.

## File Movement

Moving a database from one data center to another, or from one system to another, can be challenging given the database size, network bandwidth, and system resources available on both source and target. Whenever possible, mitigate file movement operator and downtime impact by using one of the following options:

- » Copy the files to the target location prior to plugin. Starting with Oracle Database 12c, RMAN duplicate commands are CDB and PDB-aware. [Appendix B](#) provides an example using RMAN duplicate for migration. Additional details and examples for other migration use cases are provided in [My Oracle Support Note 1576755.1: Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage](#).

Furthermore, it is possible to apply RMAN backup incrementals to initial duplicates minimizing the overall downtime. Some options include using RMAN backup incremental and apply for same-source and target platforms, or using the steps described in [Performing Cross-Platform Transport of Tablespaces Using Inconsistent Backups](#) for cross-platform migrations. For releases prior to Oracle Database 12c, you can use the steps in [My Oracle Support Note 1389592.1: Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backups](#). A small amount of additional time is required for the unplug/plugin operation to apply the final incremental; however, the source files remain untouched, allowing for immediate fallback should problems arise.

- » Configure storage so that datafiles are accessible to both source and destination environments. If the source datafiles are on shared storage accessible to the destination CDB there is no need to move the files and the plugin command can be simplified. Using the source files in place provides the fastest unplug/plugin operation; however, this modifies the source files. If fallback to the previous environment is required, a database restore and recovery is required, or at least an RMAN switch to copy with data loss.

With either of these options the NOCOPY clause can be specified in the plugin statement.

## Multitenant and Data Guard

### Operation

Data Guard physical standby databases and redo apply will normally expect a new PDB's data files to have been pre-copied to the standby site and be in such a state that redo received from the primary database can be immediately applied. The standby database also ignores any file name conversion specification on the CREATE PLUGGABLE DATABASE statement and relies solely on the standby database's initialization parameter settings for DB\_CREATE\_FILE\_DEST and DB\_FILE\_NAME\_CONVERT for locations and file naming.

Two My Oracle Support Notes have been created to assist with these nuances to help in reducing any downtime that may occur or any potential impact to disaster recovery protection of the new PDBs.

- [My Oracle Support Note 1576755.1: Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage](#)

As mentioned above, this note provides methods for migrating non-CDBs to multitenant with minimal actual downtime. In addition, the note details how to ensure that as soon as a new PDB is created on a primary CDB in a Data Guard environment, any physical standby databases will also fully create the PDB and provide immediate disaster recovery capabilities.

The methods described in this note are best used when immediate disaster recovery protection is desired AND one of the following:

- The source database is a non-CDB, can be either a 12c database or a release prior to 12c
  - The source database is a PDB that is being unplugged from one container database and plugged into another.
- [My Oracle Support Note 1916648.1: Making Use of the STANDBYS=NONE Feature with Oracle Multitenant](#)

This note describes usage of the STANDBYS=NONE clause on a CREATE PLUGGABLE DATABASE statement, also known as deferring recovery of a PDB. When the STANDBYS=NONE clause is specified on the CREATE PLUGGABLE DATABASE statement, the PDB is created at all standbys, but only enough of the PDB is created to allow redo apply to proceed. The files for the new PDB are marked as OFFLINE/RECOVER and any additional redo for that PDB will be ignored. At some point in the future, the expectation is that the PDBs files will be copied to any/all physical standby databases and recovery for the PDB be enabled and redo applied to that PDB from that point forward. Note that at PDB creation time it is not possible to specify that the PDB is created on a subset of standby databases.

If STANDBYS=NONE is included on the CREATE PLUGGABLE DATABASE statement, recovery of the new PDB will be deferred at all physical standbys and each standby on which you wish to enable recovery will require that the enable recovery process be performed. All existing PDBs on the standby databases will retain their current recovery status (DEFERRED or ENABLED).

The note describes the steps for enabling recovery of the PDB and how Data Guard role transitions will behave when having PDBs with deferred recovery.

The methods described in this note are best used when immediate disaster recovery protection is not required OR one of the following:

- The source PDB is being cloned from another container database either by remote cloning (CREATE PLUGGABLE DATABASE PDBCLONE FROM PDBSOURCE@DB\_LINK) or using a manifest file (CREATE PLUGGABLE DATABASE PDBCLONE AS CLONE USING 'manifest.xml').
- The source PDB and cloned PDB will reside in the same container database but you do not have an Active Data Guard license.

#### Standby Database Behavior when Creating, Cloning or Plugging in a PDB

The following items describe the various behaviors that will occur at the standby database when a PDB is created, either as a new PDB, a plugged in PDB or a cloned PDB. For more details see [My Oracle Support Note 2049127.1: Data Guard Impact on Oracle Multitenant Environments](#).

- Creating a new PDB from SEED

Creation of a brand new empty pluggable database will automatically replicate to each physical standby database where redo apply will copy the PDB\$SEED files at the standby site to create the new PDB.

For these cases Oracle recommends that recovery not be deferred.

- Cloning an existing PDB to a new PDB in the same container database

Behavior on a standby database differs if the standby is running Active Data Guard (the standby database is open read only with redo apply running) versus when Active Data Guard is not used.

- Cloning a PDB in the same container at the primary:
  - On any physical standby database running Active Date Guard the standby will copy the source files located at the standby site and automatically create the PDB on the standby.

For these cases Oracle recommends that recovery not be deferred.

- On any physical standby database not running Active Data Guard, the redo apply process currently cannot copy the source PDB files directly as it cannot guarantee file consistency.

For these cases, Oracle recommends deferring recovery of the PDB using the STANDBYS=NONE clause on the CREATE PLUGGABLE DATABASE statement. Recovery of the new PDB can be enabled at some point in the future once the data files of the newly plugged in PDB have been copied from the primary CDB to the standby database in a manner similar to that documented in [Note 1916648.1](#).


- Plugging in a PDB (not cloning) from another CDB

Redo apply has the ability to search in “known locations” (those derived from either DB\_FILE\_CREATE\_DEST or DB\_FILE\_NAME\_CONVERT) but only for the resulting files that it will apply redo to, not the source files. These files must be in the exact same state as the primary sees them when they are plugged in which means they must be copied from the source database to the standby database known location prior to PDB plugin at the primary. Redo apply will look in the known location for the data files and validate file header information to ensure the files are correct. For Oracle Managed Files, redo apply has been coded in such a way that it will search the known location directory and ignore file names, looking at file header information for matches.

For these cases Oracle recommends that recovery not be deferred. You should pre-copy the new PDB's data files to the known location before executing the plugin statement at the primary CDB so that redo apply can ingest them into the standby database in a manner similar to the process documented in [Note 1576755.1](#).

- Cloning a PDB from another CDB via a Manifest File

Cloning a PDB using a manifest file is similar to plugging in a PDB (previous item) but there's a difference. The process creates a new PDB which generates a new GUID for the PDB. There is no way of determining what that GUID will be prior to the new PDB being created. In an OMF/ASM configuration, this means you will not be able to pre-copy the PDB's data files to the standby “known location” mentioned earlier as the GUID is part of the known location directory structure.



For these cases, Oracle recommends deferring recovery of the PDB using the STANDBYS=NONE clause on the CREATE PLUGGABLE DATABASE statement. Recovery of the PDB can be enabled at some point in the future once the new PDB's data files have been copied from the primary database to the standby database in a manner similar to that documented in [Note 1916648.1](#).

- Performing a remote PDB clone via Database Link

Performing a remote PDB clone via a database link is similar to cloning via a manifest. For OMF/ASM configurations, the GUID of the new PDB cannot be pre-determined so it's not possible to pre-copy the data files to the "known location".

For these cases, Oracle recommends deferring recovery of the PDB using the STANDBYS=NONE clause on the CREATE PLUGGABLE DATABASE statement. Recovery of the PDB can be enabled at some point in the future once the PDB's data files have been copied from the primary database to the standby database in a manner similar to that documented in [Note 1916648.1](#).

#### Migration

When performing a migration to a pre-configured Gold or Platinum architecture that contains Data Guard, you must also ensure that copies of your datafiles are available on the standby location as part of the plugin process. The files must be in place on the standby prior to performing the plugin operation to allow media recovery to automatically find the files and continue applying redo without interruption. Oracle Database will automatically search for the files using the DB\_FILE\_NAME\_CONVERT setting or under DB\_FILE\_CREATE\_DEST directory.

If you plan to make a number of migrations to the same CDB, it may be best to defer redo transport to the standby site and reinstantiate the standby after all migrations are complete to simplify the process. This allows you to do a single step copy of all files to the standby site rather than performing copies for each plugin as they occur. The drawback to this option is that the primary database is not fully protected until the PDB standby instantiation is complete.

## Oracle Resource Management

All databases, CDBs, and PDBs will compete for the CPU, I/O, and memory resources within a hardware pool. This section describes how to plan, configure, and manage resources in a consolidated multitenant architecture environment. Existing Oracle Database management tools are all multitenant architecture-aware and should be used to ensure that a database or PDB is guaranteed a sufficient amount of resources and does not have a detrimental impact to other applications. This is especially relevant in a DBaaS environment where customers are paying for a specific service or access to a fixed amount of system resources.

Resource Management tools can be used on both non-Exadata and Exadata systems with the exception of Exadata I/O Resource Manager and Network Resource Manager. See the [Master Resource Manager Support Note 1339769.1](#) for the latest recommended patches, monitoring scripts, and step-by-step implementation guides.

#### Planning

The first step to planning consolidation with respect to performance is to determine the database's performance requirements. Figure 5 describes the performance isolation tiers: PLATINUM, GOLD, SILVER, and BRONZE. As the level of performance isolation using resource management is reduced, the consolidation density will increase.



Figure 5: Consolidation and Performance Isolation Tiers

In general, a database or PDB that has Gold tier performance requirements also has Gold tier HA service requirements, but this does not need to be strictly enforced. Databases at higher performance isolation tiers command a higher share of guaranteed resources. Resource Management tools like CDB resource management for CPU and Exadata IORM allow resources to be shared while guaranteeing each database a designated amount of resources. For these tools, databases and PDBs at lower performance isolation tiers with small resource allocations may experience performance swings on a system where the available resources are constantly changing. To provide more consistent performance use Resource Manager limits to cap the resources. Resource limits can also be an effective way to encourage customers in a DBaaS environment to pay to upgrade to a higher tier.

The simplest plan tends to be the best plan.

- » All databases and CDBs running in the same hardware pool should be in the same HA service and performance isolation tier, that is, BRONZE, SILVER, GOLD or PLATINUM. In addition, PDBs within a CDB should also have the same HA service level and performance isolation tier.
- » Use Instance Caging, memory parameters, and Exadata IORM to manage the databases and CDBs that share a hardware pool.
- » Create separate CDBs when different database versions are required, but limit the number of Oracle database homes to a maximum of 5 to reduce operational costs.
- » Create separate CDBs for different application types such as OLTP and DW. This grouping allows for easier management of the CDB because the resource requirements for OLTP and DW tend to be very different. For example, OLTP databases benefit from large buffer caches and flash caches while data warehouses benefit from high disk I/O throughput.
- » When using Oracle Exadata Database Machine, segregate applications into CDBs that benefit from flash cache versus those that can perform acceptably without it. This will allow for better flash hit ratios for OLTP databases (PDBs) that need to take advantage of Exadata Smart Flash Cache or Smart Flash Log. For example, the many test and development databases that can tolerate higher response times can be plugged into a CDB for which flash cache is disabled through the Exadata IORM plan. Critical databases for I/O sensitive applications can be plugged into a separate CDB for which flash is enabled through the Exadata IORM plan. Another example is a pure DW or analytic database that may not need Flash Cache but will most benefit from Exadata's smart offload capabilities and storage indexes.

This additional configuration recommendation is most relevant for the Bronze tier where you may have databases and applications with large varying performance requirements. For higher tiers where all applications are critical

and performance sensitive, Exadata smart flash cache dynamically adjusts for all applications' I/O requests (log writes, OLTP or small I/O reads and writes, and table scans).

- » Use CDB Resource Management from the beginning to manage the PDBs that share a CDB. This ensures that customers get access to the same set of resources and experience consistent performance, regardless if there are 10 or 100s of databases running in the hardware pool. By using it from the start, your CDB Resource Plans allow you to allocate guaranteed CPU and disk I/O resources to each PDB. In addition, the CDB Resource Plans allow you to place a hard limit on the amount of CPU or disk I/O a PDB can use. This is useful in multitenant architecture environments where customers pay for performance.
- » Focus on the Bronze tier with the highest potential consolidation density and phase in a small batch of databases at a time to gain experience and expertise with multitenant architecture.

## Controlling Resources

Table 4 summarizes the recommendations for controlling shared resources in a consolidated or multitenant architecture environment. The recommendations can be applied on all HA service level and performance tiers. Tier-specific recommendations are in a subsequent table.

TABLE 4: GUIDELINES FOR CONTROLLING RESOURCES

Resource	Guideline
Memory	<p>Independent of the HA tier (BRONZE, SILVER, GOLD, PLATINUM), memory should never be oversubscribed. Capacity planning and trend analysis should be performed before a database is migrated into a hardware pool.</p> <ul style="list-style-type: none"> <li>» Set HugePages based on your target or current shared memory requirements.</li> <li>» Do not disable the PGA_AGGREGATE_LIMIT initialization parameter. PGA_AGGREGATE_LIMIT enforces a hard limit on PGA memory usage and is critical for avoiding excessive paging, which can lead to poor performance or an instance eviction. Do not decrease its value without also decreasing PGA_AGGREGATE_TARGET. The parameter can be dynamically adjusted.</li> <li>» Use the following formulas to calculate the initial footprint of a database. Ensure that memory is not oversubscribed by following the best practices for your performance tier in Table 5. Once PGA and per-process memory statistics for actual peak memory usage are available, you can substitute those numbers for these initial approximations.               <ul style="list-style-type: none"> <li>» OLTP databases: <math>SGA\_TARGET + PGA\_AGGREGATE\_TARGET + 4\text{ MB} * (\text{Maximum PROCESSES})</math></li> <li>» DW/BI databases: <math>SGA\_TARGET + 3 * PGA\_AGGREGATE\_TARGET</math></li> </ul> </li> <li>» Set LOG_BUFFER to a maximum of 256 MB for 64-bit systems</li> <li>» References:               <ul style="list-style-type: none"> <li>» Refer to MOS 361468.1 and MOS 401749.1 for more details on HugePages</li> </ul> </li> </ul>
CPU	<p>Instance Cage databases and CDBs running on the same server to avoid CPU contention between the instances. Instance Caging prevents runaway queries and other workload surges from consuming all available system resources.</p> <ul style="list-style-type: none"> <li>» Configure Instance Caging according to the best practices for your performance tier in Table 5.</li> <li>» To configure Instance Caging, set CPU_COUNT in the 'spfile' to the maximum number of CPU threads the instance can use (with a minimum of 2). Then enable Instance Caging by setting a valid RESOURCE_MANAGER_PLAN, such as DEFAULT_PLAN for non-CDBs and DEFAULT_CDB_PLAN for CDBs. If resource requirements change, the CPU_COUNT parameter can be dynamically changed and the Instance Cage size will be immediately adjusted.</li> <li>» References               <ul style="list-style-type: none"> <li>» My Oracle Support Note 1362445.1 – 'Configuring and Monitoring Instance Caging'</li> </ul> </li> </ul>



- » See ['Best Practices for Database Consolidation On Exadata Database Machine'](#) for general best regarding configuration of OS semaphores – Exadata only.
- » Use CDB Resource Manager to manage CPU usage and contention between PDBs sharing a CDB. In a CDB Resource Plan, you can configure the SHARES and UTILIZATION\_LIMIT directives for each PDB as follows.
- » Use SHARES to configure 'Fairness'. Shares represent the relative importance of the PDBs. A PDB with more shares will be allowed to use more CPU. If all PDBs are equal, then set their shares to the same value or use DEFAULT\_CDB\_PLAN. If one PDB is twice as important as another, then double its shares. You can use the "default directive" to give PDBs a default number of shares.  
  
When migrating existing databases to a CDB, convert the individual database's CPU\_COUNT to SHARES in order to create an equivalent CDB Resource Plan.
- » Use UTILIZATION\_LIMITS to configure 'Pay for Performance'. In public clouds, limit the amount of CPU a PDB can use, based on the subscription rate. A limit prevents a "BRONZE" PDB from utilizing all the CPUs on an idle system, just as Instance Caging prevents a database from utilizing all the CPUs on an idle system. Setting DBRM 'utilization limits' from the time that the PDB is plugged in helps to ensure that expectations are properly set from the beginning.
- » If Instance Caging is enabled for a CDB, then apply the utilization limit on top of the Instance Cage size (i.e. cpu\_count) to determine the maximum amount of CPU a PDB can utilize.
- » In a RAC database, the same CDB Resource Plan must be set for all instances. If not, both Parallel Statement Queuing and Exadata IORM will operate in an unpredictable manner.
- » References
  - » My Oracle Support Note 1567141.1 - 'Migrating Databases using Instance Caging to a CDB'

**Processors and Sessions**

In most cases, PDBs use the initialization file parameter settings set at the CDB level, but a number of parameters can be set at the PDB level. Note that these settings are not stored in the parameter file; they are stored in internal tables in the CDB. For a complete list all PDB specific settings that can be modified, execute the following statement:

```
SQL> SELECT NAME FROM V$SYSTEM_PARAMETER  
WHERE ISPDB_MODIFIABLE=TRUE'  
ORDER BY NAME;
```

The following list describes the settings for both PDBs and CDBs that require changes when adding PDBs to a CDB.

- » SESSIONS:  
  
To control the number of sessions available for a CDB and its PDBs, you set the total number of SESSIONS for the CDB in the initialization parameter file at root level. For each PDB, you have the option of setting a value for the SESSIONS parameter to pose a hard limit for the number of sessions that PDB can start. The initialization parameter SESSIONS sets the maximum for the CDB, the SESSIONS setting within the PDB is specific to that PDB.
- » PARALLEL\_MAX\_SERVERS:  
  
PARALLEL\_MAX\_SERVERS specifies the maximum number of parallel execution processes and parallel recovery processes for an instance. As demand increases, an Oracle database increases the number of parallel execution processes up to PARALLEL\_MAX\_SERVERS. Ensure that each application can meet its performance requirements with the lower value. If PARALLEL\_MAX\_SERVERS is set too high, then memory resource shortages may occur during peak periods, which can degrade performance and destabilize the database node.  
  
On Exadata:
  - » For X2-2, X3-2, X4-2, X5-2, X6-2: sum(PARALLEL\_MAX\_SERVERS) for all instances and PDBs <= 240.
  - » For X2-8, X3-8, X4-8, X5-8, X6-8: sum(PARALLEL\_MAX\_SERVERS) for all instances and PDBs <= 1280In a CDB, parallel servers are shared by all PDBs. To limit the maximum number of parallel servers that a PDB can use at any time, set the PARALLEL\_SERVER\_LIMIT directive in the CDB Resource Plan to the maximum percentage of PARALLEL\_MAX\_SERVERS that a PDB can use at any time.
- » Limit the number of processes and connections to the database servers:  
  
Having an appropriate number of processes has many advantages such as avoiding or reducing memory, CPU and





	<p>database latch contention, log file sync waits, and overall application failover times. The reduced process and connection count can also lead to better performance and throughput but most importantly system stability.</p> <p>Use a conservative active process count to a maximum of 2 times CPU cores and a total process count for the entire database node to be maximum 10-12 times CPU cores. By using one or more of the following techniques, you can lower the overall process count and improve performance and stability:</p> <ul style="list-style-type: none"><li>» Limit the number of processes and connections to the database servers by using connection pools and setting the maximum number of connections to a value slightly above estimated active working sessions. Avoid the expensive allocation and de-allocation of processes by eliminating dynamic connection pools by setting min=max connections to be the same.</li></ul> <p>Note: if running on Exadata, refer to <a href="#">Exadata Consolidation Best Practices</a> for suggestions on using shared servers/MTS, connection pools, etc.</p> <ul style="list-style-type: none"><li>» Configure Oracle listeners to throttle incoming connections to avoid logon storms after a database node or instance failure.</li></ul> <p>» Limit Redo Apply Parallelism if this is Active Data Guard</p> <p>» References</p> <ul style="list-style-type: none"><li>» See 'Best <a href="#">Practices for Database Consolidation On Exadata Database Machine</a>' for more information on reducing process count – Exadata only</li></ul>
IORM (Exadata Only)	<ul style="list-style-type: none"><li>» Enable Exadata IORM to ensure fair access to the disks across all CDBs and PDBs. By default, IORM operates with the 'basic' IORM objective and performs only a nominal amount of scheduling to keep latencies for critical disk I/Os from reaching extreme levels. To fully enable, set the 'IORM objective' to 'auto' instead of the default 'basic'. With objective, IORM enforces the inter-database IORM plan and all CDB and database resource plans.</li><li>» When OLTP databases and data warehouses share the same Exadata storage cells, most OLTP I/Os will be serviced from flash and most data warehouse I/Os will be serviced from disks. If the OLTP databases are issuing I/Os to disks, contention from the data warehouse workloads may cause unacceptable increases in disk latencies. By enabling IORM, you can lower the disk latencies for OLTP I/Os by providing large resource allocations to the OLTP databases and PDBs in the inter-database IORM plans and CDB Resource Plans. If the disk latencies continue to be too high or inconsistent, you can modify the IORM objective to "balanced" or "low latency".</li><li>» When consolidating many applications that do smart scans, configure UTILIZATION_LIMIT in the inter-database, CDB, or database resource plan.</li><li>» Segregating different workloads into different CDBs helps to ensure data warehouse applications do not consume flash resources from critical OLTP databases, allowing the OLTP applications to maintain their required SLAs.</li><li>» References: See My Oracle Support 1363188.1 – 'Configuring Exadata I/O Resource Manager for Common Scenarios'</li></ul>
Network	<ul style="list-style-type: none"><li>» Network resource management (Exadata-only) automatically and transparently prioritizes critical database network messages ensuring fast response times for latency critical operations. Prioritization is implemented in the database, database InfiniBand adapters, Oracle Exadata Storage Server Software, Exadata storage InfiniBand adapters, and InfiniBand switches to ensure that prioritization happens through the entire InfiniBand fabric. Latency sensitive messages such as Oracle RAC Cache Fusion messages are prioritized over batch, reporting, and backup messages. Log file write operations are given the highest priority to ensure low latency for transaction processing. Network resource management is always on and available starting Exadata 11.2.3.3.0 with IB switch software release 2.1.3-4 on DB version 11.2.0.4/12c.</li><li>» Avoid excessive Cache Fusion network traffic by activating a PDB on a minimum number of instances in a RAC database. Use PDB services to determine the instances where a particular PDB is active to avoid needless distribution of data. In addition, be sure that the load is balanced across all instances of a CDB RAC database.</li></ul>
Storage Grid	<ul style="list-style-type: none"><li>» Configure one shared storage Grid for each Hardware Pool. Managing one shared storage Grid is simpler with lower administrative costs. Space and bandwidth utilization are also more efficient with shared storage.</li><li>» If this is a GOLD or PLATINUM Hardware Pool, use ASM high redundancy for the DATA and RECO disk groups for best data protection and redundancy during Exadata cell rolling upgrade and from storage type failures – Exadata only.</li><li>» Since a hardware pool is designated to one tier, the shared storage GRID should only be servicing one tier.</li></ul>

<b>Cluster</b>	<ul style="list-style-type: none"> <li>» Use one cluster per Hardware Pool. All database services are managed by one Oracle Clusterware installation which should be used to further load balance and route applications to specific database instances or PDBs in the cluster.</li> <li>» With Oracle Multitenant, it's recommended to keep the number of CDBs to a maximum of 10 per cluster which implies maximum 10 instances per database node. Each CDB instance can have a maximum of 252 PDBs.</li> </ul> <p>The actual number of database instances or PDBs you create per database node or cluster depends on application workload and system resource consumption of each instance/PDB.</p>
----------------	--

Guidelines for configuring databases based on their performance tier are provided in Table 5.

TABLE 5: RESOURCE MANAGEMENT GUIDELINES BY PERFORMANCE TIER

Performance Tier	CPU	Memory	Exadata I/O
<b>Platinum</b>	<p>Instance Cage with no over-subscription. The sum of CPU_COUNT across all databases should not exceed 75% of the server's CPU threads.</p> <p>Consider binding each instance to dedicated CPUs or NUMA nodes by configuring Linux cgroups or Solaris Resource Pools and setting the PROCESSOR_GROUP_NAME parameter. See MOS note 1585184.1.</p>	<p>Configure SGA_TARGET and PGA_AGGREGATE_TARGET for all databases.</p> <p>The sum of memory for PGA, SGA and client processes should not exceed 75% of the total system memory.</p> <p>Beware of consolidating with other 11g databases. Without the PGA_AGGREGATE_LIMIT parameter, which is only available in Oracle 12c, it is not possible to strictly restrict a database's PGA usage.</p>	<p>Configure inter-database IORM. If storage cells host databases from multiple performance tiers, give larger allocations or shares to databases at higher tiers.</p>
<b>Gold</b>	<p>Instance Cage with no over-subscription. The sum of CPU_COUNT across all databases should not exceed 90% of the server's CPU threads.</p>	<p>Configure SGA_TARGET and PGA_AGGREGATE_TARGET for all databases.</p> <p>The sum of memory for PGA, SGA and client processes should not exceed 75% of the total system memory.</p>	<p>Configure inter-database IORM. If storage cells host databases from multiple performance tiers, give larger allocations or shares to databases at higher tiers.</p>
<b>Silver</b>	<p>Instance Cage. Moderate amount of over-subscription is acceptable if the databases' workloads peak at different times.</p>	<p>Configure SGA_TARGET and PGA_AGGREGATE_TARGET for all databases.</p> <p>The sum of memory for PGA, SGA and client processes should not exceed 80% of the total system memory.</p>	<p>Configure inter-database IORM. If storage cells host databases from multiple performance tiers, give larger allocations or shares to databases at higher tiers.</p>
<b>Bronze</b>	<p>Instance Cage. Over-subscribe.</p>	<p>Configure SGA_TARGET and PGA_AGGREGATE_TARGET for all databases.</p> <p>The sum of memory for PGA, SGA and client processes should not exceed 90% of the total system memory.</p>	<p>Configure inter-database IORM. If storage cells host databases from multiple performance tiers, give larger allocations or shares to databases at higher tiers.</p> <p>Consider setting a "limit" on Bronze databases for more consistent performance and to constrain its smart-scans.</p> <p>Consider disabling flash cache via the inter-database plan for non-critical databases such as test and dev if the flash cache is not large enough to accommodate all databases. See the next section for monitoring the flash cache miss rate.</p>

## Monitoring Resources

Monitoring and analyzing database performance starts before the database is migrated to the hardware pool. It is important to understand the database's average and peak resource consumption at the source before deciding if it is a good candidate for consolidation.

After migration to the target consolidated environment, close monitoring of performance indicators remains important to see if the consolidation plan was viable. It is always recommended to use a test environment first so that the process and impact for each database that will be migrated can be validated before going into production.

Monitoring should occur at 3 levels:

- » System monitoring. The servers and storage should be monitored to see if CPU, memory, and storage utilization are at acceptable levels. If you are planning to consolidate more databases onto the servers or storage you should also monitor the available capacity or headroom.
- » CDB or database monitoring. The database should be monitored to see how many resources it is actually using. This check is particularly important for resources that cannot be constrained using Resource Manager, such as PGA. The database should also be monitored to see how much Resource Manager is restricting or throttling it. If you see waits for CPU or I/Os due to Resource Manager and the database's performance is not acceptable, then you can either tune Resource Manager or move the database to a system with more resources.
- » PDB monitoring. Each PDB should be monitored in the same way as a database. If its actual resource usage surpasses the resources guaranteed by the current CDB Resource Plan, then its performance will probably degrade as more PDBs are added to the CDB. If performance degradation is not acceptable then do not add more PDBs to this CDB instance.

Key performance indicators are highlighted in the table below.

TABLE 6: KEY PERFORMANCE INDICATORS

Monitor /Administer	Guideline
Memory	<p>For the system:</p> <ul style="list-style-type: none"><li>» No paging should be seen. Use the vmstat command to monitor for zero or very low page-in and page-out rates.</li><li>» Memory should never be oversubscribed. Use /proc/meminfo on Linux to obtain the total system memory. Compare it to the memory actually allocated for the different Oracle databases and client processes. SGA usage can be computed from each database's SGA_TARGET parameter. PGA usage can be determined, using the statistics below.</li></ul> <p>For the database or CDB:</p> <ul style="list-style-type: none"><li>» Monitor actual PGA usage per instance via v\$pgastat. The "maximum PGA allocated" statistic is the maximum PGA that the instance has allocated since its startup. The statistic "total PGA allocated" is the amount of PGA that the instance has allocated currently. Monitoring these values enables the DBA to know how much PGA is actually being used by the instance. They should be compared to PGA_AGGREGATE_TARGET to see if the database is exceeding it and by how much.</li></ul>
CPU	<p>For the system:</p> <ul style="list-style-type: none"><li>» Monitor the system's CPU utilization, using OS tools or the "Host CPU Utilization" metric from v\$sysmetric_history. If the CPU utilization is near 100%, use OS tools or the "OS Load" metric from v\$sysmetric_history to determine how over-subscribed the system is. Excessive over-subscription should be avoided, as specified in Table 5, Resource Management Guidelines by Performance Tier. Use Instance Caging to avoid excessive loads.</li></ul>



	<p>For the database or CDB:</p> <ul style="list-style-type: none"><li>» If Instance Caging or CDB Resource Manager is enabled, monitor the amount of CPU that the instance actually used and the amount of CPU it needed but was prevented from using. Use <code>v\$srcmrgmetric_history</code>, as described in MOS note 1338988.1. The sum of 'avg_running_sessions' across all Consumer Groups and PDBs specifies the number of CPUs actually used. The sum of 'avg_waiting_sessions' across all Consumer Groups and PDBs specifies the throttling performed by Resource Manager due to insufficient CPU. It corresponds to the additional number of CPUs needed.</li><li>» For CDBs, monitor the available CPU capacity or headroom to determine if more PDBs can be added. If CDB Resource Manager is enabled, calculate the total amount of CPU needed by summing 'avg_running_sessions' and 'avg_waiting_sessions' from <code>v\$srcmrgmetric_history</code> across all Consumer Groups and PDBs. If this sum is less than <code>CPU_COUNT</code>, then the gap between this sum and <code>CPU_COUNT</code> is the CPU headroom on the CDB and you can consider adding more PDBs. If this sum is greater than <code>CPU_COUNT</code>, then this CDB is already operating at capacity and has no headroom. Adding more PDBs will only increase contention for CPU. Therefore, additional PDBs should only be added for BRONZE CDBs if the existing PDBs and their applications can tolerate some drop in performance. Monitoring should be done during peak hours. For more details, see MOS note 1338988.1.</li></ul> <p>For the PDBs:</p> <ul style="list-style-type: none"><li>» Monitor the actual CPU usage and CPU wait time of each PDB. Use <code>v\$srcmrgmetric_history</code> to see how many CPUs a PDB actually used, using the 'avg_running_sessions' metric. Compare the actual CPU usage with the PDB's guaranteed CPU, which is based off the PDB's shares divided by the total number of shares across all PDBs that are open on the instance. For example, if the PDB has 1 share and the total shares is 5, then the PDB is guaranteed 1/5th of <code>CPU_COUNT</code>. If the PDB's actual CPU usage is above its guaranteed CPU usage, then the PDB's performance may suffer if additional PDBs are added. Use <code>v\$srcmrgmetric_history</code> to view the PDB's CPU wait time, using the 'avg_waiting_sessions' metric. If non-zero, the PDB's performance could be improved by increasing its shares or utilization limit in the CDB Resource Plan. For more details, see MOS note 1338988.1.</li></ul>
Disk/Flash	<p>For Exadata storage cells. Monitor Exadata I/O metrics using the script in MOS note 1337265.1. In most cases, these metrics can also be viewed using Enterprise Manager 12c.</p> <ul style="list-style-type: none"><li>» Monitor the disk latency for OLTP I/Os such as buffer cache reads using the <code>CD_IO_ST_RQ</code> metric. If the latency is high and is causing performance problems for OLTP databases, enable Exadata IORM. In the inter-database IORM plan, give large allocations to OLTP databases from high performance tiers. If the latency is still not acceptable, then consider changing the IORM objective from "auto" to "balanced" or "low latency".</li><li>» Monitor the disk load using the <code>CD_IO_LOAD</code> metric. A value above 5 is considered high for systems with OLTP databases since higher values result in increased latency. If higher latencies are not acceptable, no more databases should be added. For data warehouses, higher values are fine as high loads result in improved disk throughput. However, if the load metric is consistently over 20, no more databases should be added.</li><li>» Monitor the actual disk IO utilization across databases using the <code>DB_IO_UTIL_SM</code> + <code>DB_IO_UTIL_LG</code> metrics. This comparison shows which databases are most heavily using the disks.</li><li>» Monitor flash write IOPS using <code>FC_IO_RQ_DISK_WRITE</code>. If the total write IOPS has reached the published maximum and degradation of flash performance is unacceptable, then don't add new databases or CDBs that use flash cache. Flash cache can be disabled for a database or CDB by using the "flashCache=off" directive in the inter-database IORM plan.</li><li>» Compare <code>FC_IO_RQ_R</code> with <code>FC_IO_RQ_R_MISS</code> to calculate the OLTP flash hit ratio. If the current performance should remain the same, make sure the flash has headroom by monitoring its flash hit ratio before adding more databases to use flash. For example, if performance is acceptable at an 80% or higher flash hit ratio, performance may deteriorate if more databases are added and the flash hit ratio drops below 80%.</li></ul> <p>For the database or CDB:</p> <ul style="list-style-type: none"><li>» Monitor I/O wait events, using AWR. If the problematic wait event is "db file sequential read", then monitor and tune the database's flash cache hit rate (see above), the storage cell's disk latency (see above), and the database's IORM throttle time for small requests (see below). If the problematic wait event is "cell smart table scan", then monitor and</li></ul>



	<p>tune the database's throttle time (see below).</p> <ul style="list-style-type: none"> <li>» If IORM is enabled, monitor the average IORM throttle time per request, using the DB_IO_WT_SM_RQ and DB_IO_WT_LG_RQ metrics. If the wait times are large and the database performance is unsatisfactory, either the database's allocation/share or utilization limit in the inter-database plan needs to be increased. If the wait times are large across all databases, then the disks have reached their maximum capacity. In this case, no new databases should be added unless the existing databases can tolerate a drop in performance.</li> <li>» For CDBs, monitor the actual disk IO utilization across PDBs using the PDB_IO_UTIL_SM + PDB_IO_UTIL_LG metrics. This comparison shows which PDBs within the CDB are most heavily using the disks.</li> </ul> <p>For the PDB:</p> <ul style="list-style-type: none"> <li>» If IORM is enabled, monitor the average IORM throttle time per request, using the PDB_IO_WT_SM_RQ and PDB_IO_WT_LG_RQ metrics. If the wait times are large and the PDB performance is unsatisfactory, either the PDB's share or utilization limit in the CDB Resource Plan needs to be increased. If the wait times are large across all PDBs, then the allocation and/or limits for the CDB in the inter-database IORM plan need to be increased. Until then, no new PDBs should be added unless the existing PDBs can tolerate a drop in performance.</li> </ul>
Database performance indicators	<ul style="list-style-type: none"> <li>» Use Automatic Workload Repository and/or Enterprise Manager 12c for the database load profile, drill down to see top wait events. From here drill down into specific waits.</li> <li>» Use ASH analytics to view breakdown of CPU usage by consumer group or PDB</li> </ul>
Application performance indicators	<ul style="list-style-type: none"> <li>» Use Automatic Workload Repository and/or Enterprise Manager 12c for the database load profile, drill down to see top wait events. From here drill down into specific waits.</li> <li>» Use ASH analytics to view breakdown of CPU usage by consumer group or PDB</li> </ul>
General	<ul style="list-style-type: none"> <li>» Exachk in MOS note 1070954.1 (Exadata only)</li> </ul>

## High Availability and Data Protection

Service level expectations in a non-consolidated environment are quite different compared to a consolidated environment. Take for example a standalone database used by a developer or a department. The level of disruption caused by a database down event is limited to a smaller user community that can often find other work to remain productive while the outage is resolved. Now consider what happens when this same database is consolidated with 100 other databases, each supporting different departments and user communities. The level of disruption to the enterprise of an outage that impacts the consolidated database is magnified by a 100 times, making HA and data protection a much higher priority.

Oracle Multitenant uses the Oracle Maximum Availability Architecture (MAA) to addresses the HA and data protection requirements of consolidated environments. In addition to MAA's customary objectives for HA and data protection there are objectives that are specific to a multitenant architecture context:

- » Manage-as-One Simplicity. MAA best practices must easily scale the management of large consolidated environments so that HA and data protection objectives are achieved while realizing maximum cost benefit (capital and operating costs).
- » Isolation. MAA best practices must prevent problems that impact a single PDB from impacting the availability of other PDBs in the CDB.

---

*On the surface the first two objectives appear contradictory. Isolation for “n” environments often results in “n” different environments that must be individually managed. Oracle MAA can leverage the multitenant architecture so that HA is achieved with minimal compromise in isolation while achieving substantial benefits from being able to manage-as-one.*

---

- » Oracle MAA enables the management of a CDB as a single database regardless of how many PDBs it contains. A CDB with 100 PDBs can achieve 100 to 1 reduction for many maintenance and operational tasks: a single RMAN backup, a single Oracle Active Data Guard standby for disaster recovery, and a single database to upgrade or patch. Oracle MAA also provides the flexibility to manage a PDB in isolation from other PDBs when appropriate. For example:
- » If an individual PDB must be restored, RMAN can do so without impacting other PDBs that are open in the same CDB. Note that a PDB that has just been plugged in should be backed up immediately after the plugin operation to ensure that it can be recovered should a problem arise before the next regularly scheduled CDB backup.
- » If fast point-in-time recovery is required, the first release of Oracle Multitenant enables using Flashback Database at the CDB level. A future release of Oracle Multitenant is planned to enable Flashback Database to be used on an individual PDB without impacting the availability of other PDBs.
- » If an individual PDB experiences a problem that the administrator believes has the potential to impact other PDBs in the CDB, the suspect PDB can be easily unplugged and plugged into another CDB where the issue can be resolved in isolation. An unplug/plugin operation also enables the flexibility of applying a patch without impacting other PDBs running in the original CDB. Once the problem is resolved you can leave the PDB in the new CDB, or you can unplug/plugin the PDB back into its original CDB after the new patches have been installed.
- » If one PDB is corrupted on the primary CDB database and there's an existing standby database where the PDB is healthy, you can execute a single PDB failover by unplugging and plugging the PDB from the standby to a new CDB. Refer to My Oracle Support Note: Unplugging a Single Failed PDB from a Standby Database and Plugging into a New Container (Doc ID 2088201.1).
- » Oracle Resource Manager will prevent a PDB from consuming more than its assigned share of system resources if there is a sudden spike in workload, a bug, or some other event that changes consumption patterns.
- » Oracle GoldenGate logical replication can be used to replicate one or more PDBs within a CDB. This provides the flexibility to migrate a PDB to another CDB with minimal or zero (using bi-directional replication) downtime. The process begins by creating a clone of the PDB in a new CDB to support whatever is intended for planned maintenance (platform migration, Oracle Database upgrade, application upgrade that modifies back-end database objects, or other database maintenance that would normally require downtime). When the cloned PDB is operational at the new version, GoldenGate replication is used to synchronize it with new transactions that have occurred since the original source was cloned. Users are transitioned to the new version once validation is complete. There are two options for handling this transition:
- » Minimal downtime can be achieved by terminating user sessions, allowing Oracle GoldenGate to finish replicating all committed transactions, shutting down the source database, and allowing users to connect to the new version.
- » Zero downtime can be achieved using bi-directional replication. This allows users to be gradually migrated as they naturally disconnect then reconnect, creating the user experience of zero downtime. It also allows for immediate fail-back if unanticipated issues arise at the new version as workload increases. Bi-directional replication requires the administrator to implement conflict detection and resolution using basic capabilities provided with Oracle GoldenGate (first change wins) or by extending Oracle GoldenGate to implement more sophisticated conflict resolution schemes. Knowledge of the application is required. Refer to Multitenant and GoldenGate Considerations.

## Multitenant and GoldenGate Considerations

Starting with GoldenGate release 12.1.2 it is possible to replicate data out of and into PDBs. For example, it is possible to extract data from one or more PDBs and replicate to non-multitenant databases, or extract data from a non-multitenant database and replicate to one or more PDBs.

The Installation and Configuring Oracle GoldenGate for Oracle Database guide provides the configuration details for GoldenGate with a multitenant database:

[http://docs.oracle.com/goldengate/c1221/gg-winux/GIORA/config\\_containerdb.htm#GIORA942](http://docs.oracle.com/goldengate/c1221/gg-winux/GIORA/config_containerdb.htm#GIORA942)

There are some additional performance recommendations for GoldenGate and multitenant databases:

- Use a single integrated capture mode Extract to extract data from one or more PDBs. Because each Extract process reads the entire CDB redo stream which contains data from all PDBs, using a single Extract reduces the amount of additional I/O incurred when adding additional Extract processes.
- If there is a requirement to replicate data to multiple target databases, configure several Data Pumps to transmit a subset of the trail file contents to each target database host.
- A Replicat can apply data to only one PDB, dictated by the SQL\*Net connect string. It is possible for a single Replicat to apply data from multiple PDBs contained in the trail files.
- Before configuring additional integrated Replicats it is important to monitor the available CPU bandwidth on the server to make sure there is enough headroom for additional processes. Integrated Replicat uses an automatic parallelism feature which will increase the number of apply server processes when there is additional work waiting to be applied. The maximum number of apply servers created by Replicat is controlled by the GoldenGate parameter INTEGRATEDPARAMS (MAX\_PARALLELISM n). The default value is 50, and this may need to be reduced to make sure a single integrated Replicat does not consume too much CPU, starving other Replicats of required resources.

## Managing Unplanned Outages

Table 7 identifies various unplanned outages that can impact a database in multitenant architecture. It also identifies the Oracle HA solution to address that outage that is available in each of the HA tiers described earlier in this paper.

TABLE 7: UNPLANNED OUTAGE MATRIX FOR MULTITENANT ARCHITECTURE

Event	Solutions for Bronze, Silver, Gold & Platinum	Recovery Windows (RTO)	Data Loss (RPO)
Instance failure	BRONZE: Oracle Restart	Minutes if server can restart	Zero
	SILVER: Oracle RAC or optionally Oracle RAC One Node	Seconds with Oracle RAC Minutes with Oracle RAC One Node	Zero
	GOLD: Oracle RAC	Seconds	Zero
	PLATINUM: Oracle RAC with Application Continuity parameter. See MOS note 1585184.1.	Zero Application Outage	Zero

Permanent node failure (but storage available)	BRONZE: Restore and recover	Hours to Day	Zero
	SILVER: Oracle RAC	Seconds	Zero
	SILVER: Oracle RAC One Node	Minutes	Zero
	GOLD: Oracle RAC	Seconds	Zero
	PLATINUM: Oracle RAC with Application Continuity	Zero Application Outage	Zero
Storage failure	All tiers: Automatic Storage Management	Zero downtime	Zero
Data corruptions	BRONZE/SILVER: Basic protection Some corruptions require restore and recover of PDB or entire CDB	Hour to Days	Since last backup if unrecoverable
	GOLD/PLATINUM: Comprehensive corruption protection and Auto Block Repair with Oracle Active Data Guard	Zero with auto block repair Seconds to minutes if corruption due to lost writes and using Data Guard Fast Start failover.	Zero unless corruption due to lost writes
Human error	ALL: Logical failures resolved by flashback drop, flashback table, flashback transaction, flashback query and undo.	Dependent on detection time but isolated to PDB and applications using those objects.	Dependent on logical failure
	All: Comprehensive logical failures impacting an entire database and PDB that requires RMAN point in time recovery (PDB) or flashback database	Dependent on detection time	Dependent on logical failure
	GOLD/PLATINUM: With Oracle GoldenGate, you can fail over just one PDB	Dependent on detection time but actual failover can take seconds	Dependent on logical failure
Database unusable, system, site or storage failures, widespread corruptions or disasters	BRONZE/SILVER: Restore and recover	Hours to Days	Since last backup
	GOLD: Fail over to secondary (Oracle Active Data Guard or Oracle GoldenGate)	Seconds	Zero to Near Zero
	PLATINUM: Active Data Guard Failover with Application Continuity	Zero Application Outage	Zero
Single PDB unusable	BRONZE/SILVER: Restore and recover PDB	Hours to Days	Since last backup
	GOLD: Fail over to CDB secondary or perform PDB Failover ( <a href="#">Note 2088201.1</a> ) (Oracle Active Data Guard or Oracle GoldenGate)	Minutes Zero Application Outage	Zero to Near Zero Near Zero for Oracle GoldenGate
	PLATINUM: Active Data Guard Failover with Application Continuity or perform PDB Failover ( <a href="#">Note 2088201.1</a> ). (Oracle Active Data Guard or Oracle GoldenGate)	Minutes for Oracle GoldenGate and PDB failover since Application Continuity is not supported for these cases.	Zero for Data Guard and PDB failover Near Zero for Oracle GoldenGate
Performance degradation	All tiers: Database Resource Manager and Tuning	No downtime but degraded service	Zero



**Note:** To ensure that PDBs can be recovered, a PDB or CDB backup should be performed immediately after a PDB plugin operation. To restore a PDB the associated CDB must be available and operational.

While it is possible to remedy many issues with an individual PDB without impact to other PDBs, there are situations in which isolation is required. For example, you may need to apply a patch to the Oracle Home being used by the CDB. In this scenario it is recommended that you create a new Oracle Home and CDB, then unplug the problematic PDB and plug it into the new CDB. You can then resolve the problem with 100% certainty of zero impact to other PDBs in the original CDB. Once the problem is resolved you can unplug/plug the PDB back into its original CDB or leave it in the new CDB until some future point in time.

In a Data Guard environment when there are issues with any PDB at the primary site requiring failover, you would normally fail over the entire CDB with all of its PDBs to a standby site. While it is not directly possible to fail over an individual PDB to the standby, it is possible to quickly relocate a single PDB that has failed beyond immediate repair at the primary by extracting it from a standby site and plugging it into a co-located CDB at the standby site with minimal downtime. See [Note 2088201.1](#) for the steps to perform this process. The steps provide you a method for moving the impacted PDB in isolation while allowing the remaining PDBs to keep operating normally.

## Managing Planned Maintenance

From a planned maintenance perspective all of the traditional solutions available to non-CDBs will work in a multitenant architecture environment. Additionally, there are cases where the administrator can decide if the maintenance should be done to just one PDB or all PDBs in the same container. Oracle provides the flexibility to handle either situation.

Oracle HA solutions for planned maintenance are provided in table 8.

TABLE 8: PLANNED MAINTENANCE MATRIX FOR MULTITENANT ARCHITECTURE

Event	Solutions for Bronze, Silver, Gold and Platinum	Expected Downtime
Migrations	Refer to: <a href="#">Migration to Multitenant Architecture with MAA Service Tiers</a>	Varies
Dynamic and Online Resource Provisioning or Online reorganization and redefinition	ALL: Online Reorganization and Redefinition of select objects within each PDB  Documentation: <a href="#">Dynamic and Online Resource Provisioning</a> and <a href="#">Online Reorganization and Redefinition</a>	Zero
Online Patches	ALL: Entire CDB can be online patched if relevant	Zero
Database and Grid Infrastructure Patches and One-off Patches	ALL: PDB can unplug and plug into a separate CDB with targeted software release SILVER: Entire CDB can leverage Oracle RAC One Node rolling upgrade if relevant GOLD/PLATINUM: Entire CDB can leverage Oracle RAC rolling upgrade if relevant. Application continuity will complement in the PLATINUM tier. GOLD: Optionally entire CDB can leverage Data Guard standby-first patching and issue Data Guard switchover PLATINUM: Optionally entire CDB can leverage Data Guard	Estimated seconds to hour with no datafile copy option Zero by relocating services Zero by relocating services Zero application outage Seconds to minutes Zero application outage

	standby-first patching and issue Data Guard switchover and application continuity	
Database Patchsets	<p>ALL: <a href="#">PDB can unplug and plug into a separate CDB with targeted software release</a></p> <p>GOLD/PLATINUM: Entire CDB can leverage Data Guard database rolling upgrade for patchsets and major database releases</p> <p>PLATINUM: CDB or PDB can fail over to secondary GoldenGate replica residing on the targeted software version</p>	<p>Estimated seconds to hour with no datafile copy option</p> <p>Seconds to Minutes</p> <p>Zero downtime</p>
Application upgrades	<p>PLATINUM: Edition-Based Redefinition requires developers to design to leverage this feature</p> <p>PLATINUM: PDB can switch over to GoldenGate replica with the targeted application changes</p> <p>Documentation: <a href="#">Online Application Maintenance and Upgrades</a></p>	<p>Reduce software sprawl and risk. Reduce OPEX and increase overall stability and availability.</p>

## Patching and Upgrades

Patching and upgrades are the areas most affected by the multitenant architecture. With a single upgrade all PDBs in a CDB can be upgraded to the later version using the “manage many as one” capability provided by the multitenant architecture. Existing tools such as Grid Infrastructure Rolling Patching, Real Application Cluster Rolling Patching =, Data Guard Standby First Patching, Data Guard Transient Logical Rolling Upgrade, or Oracle GoldenGate can also be used to reduce downtime to near zero or zero to upgrade an entire CDB in a single operation.


If you only want to upgrade a subset of the PDBs in a CDB you can create a brand new CDB with the upgraded version and unplug/plug PDBs. This provides the flexibility you may require to deal with specific application needs. This operation can take less time than upgrading an individual database in place. Oracle GoldenGate also provides the option to upgrade PDBs with minimal or zero downtime by replicating between different CDBs.

## Life Cycle Management for DBaaS

Oracle Enterprise Manager 12c plays a major role in managing the different phases of the lifecycle of databases in a multitenant architecture and Database as a Service (DBaaS). Oracle Enterprise Manager 12c delivers self-service deployment of IT resources for business and technical users along with resource pooling models that cater to various multitenant architectures. DBaaS is a paradigm where end users (DBAs, Developers, QA Engineers, Project Leads, and so on) can request database services, consume it for the lifetime of the project, and then have them automatically de-provisioned and returned to the resource pool.

DBaaS provides:

- » A shared, consolidated platform on which to provision database services
- » A self-service model for provisioning those resources
- » Elasticity to scale out and scale back database resources
- » Chargeback based on database usage



For existing implementations where customers already have databases under operation, Oracle Enterprise Manager 12c can discover those databases automatically, baseline their usage, and provide consolidation advisory for migrating to a multitenant architecture. It can then provide a guided flow to migrate the non-CDB databases to a multitenant architecture. In the course of the migration it can also upgrade the database from an earlier version (that is, Oracle Database 10g or Oracle Database 11g).

For the rollout of new databases, Oracle Enterprise Manager 12c provides out-of-the-box workflows for provisioning PDBs leveraging the native plug and unplug mechanisms. The same procedures are exposed to the self-service interface for consumption by self-service DBaaS users.

Oracle Enterprise Manager provides the necessary automation to manage the PDBs in mass scale. The configuration management capabilities such as inventory management and reporting helps prevent any unwanted sprawl. Oracle Enterprise Manager comes with the industry's leading configuration drift management capability so that the entire stack can be compared and checked for consistency against a golden baseline. Another feature that deserves mention in this context is Compliance Management. Oracle Enterprise Manager comes with out-of-the-box rules to check the configuration of the databases against well-known industry standards and best practices.

Finally, database patching and upgrades can be a very time consuming and labor intensive activity within the data center, especially when the administrators have to deal with hundreds and thousands of PDBs and underlying CDBs. Oracle Enterprise Manager 12c comes with out-of-the-box automation for patching and upgrade, supplemented by pre-flight checks and post-patch reporting.

To summarize, the lifecycle management of databases in a multitenant architecture includes:

- » Migration from non-CDBs
- » Initial provisioning and cloning of PDBs
- » Inventory tracking, configuration drift tracking, and topology mapping
- » Configuration compliance management
- » Patch and upgrade automation

For more information, refer to:

- » [Database Management](#)
- » [Database Lifecycle Management](#)
- » [Database Cloud Management](#)

## Conclusion

Implementing database consolidation and DBaaS requires a holistic approach so that strategies designed to address one set of objectives (for example, reducing systems footprint) do not create new challenges along other dimensions of a consolidated environment (for example, performance, HA, data protection, or management costs).

Oracle Database 12c with Oracle Multitenant, Oracle MAA best practices, and Oracle Enterprise Manager provide the necessary holistic solution required for efficient, reliable database consolidation and DBaaS. While these solutions enable consolidation and DBaaS on any platform, the full realization of minimizing total life-cycle costs while providing optimal service is achieved using Oracle Engineered Systems.

## Appendix A: Exadata Consolidation Density and Performance

Oracle has conducted a series of tests that validated the ability to achieve high consolidation densities on Oracle Exadata Database Machine. Two series of tests were run on an Oracle Exadata Database Machine using an identical workload and Oracle Database configuration. The first series of tests disabled all Exadata-specific features to provide a baseline for comparison. While the intention was to simulate a non-Exadata system, this approach provided an aggressive baseline for comparison given the high I/O and network bandwidth and robust performance characteristics of an Exadata system even without Exadata features enabled. The second series of tests ran the same workload as the first on the same machine, but this time with all Exadata-only features enabled. This made it possible to measure the ability of Exadata's unique features to increase consolidation density for database workloads.

### System Configuration:

- » [Oracle Exadata Database Machine X4-2](#) full rack
- » 8 database servers with a total of 192 CPU cores and 4TB of memory
- » 14 Exadata Storage servers with 168 cores dedicated to SQL processing in the storage tier
- » 44 TB of Exadata Smart Flash Cache
- » 40 Gb/second internal InfiniBand network

### Database Configuration:

- » Oracle RDBMS 12.1.0.1 (PSU 2) and Oracle Grid Infrastructure 12.1.0.1 (PSU 2)
- » DBFS\_DG is normal redundancy (double mirror), DATA and RECO ASM disk groups are high redundancy (triple mirror)
- » Oracle single instance database (non-RAC), Oracle RAC One Node, and Oracle RAC were attempted
- » Standard Exadata configuration best practices for the Oracle Database were used, such as using HugePages, proper system and database settings, and MAA settings
- » Each OLTP database used 4 GB SGA
- » DW/Reporting database ran on all 8 nodes and had a 7 GB SGA

The following Exadata features were isolated to compare non-Exadata and Exadata consolidation densities

- » Smart Flash logging
- » Smart Flash cache
- » Flash cache compression
- » Network resource management
- » Offload scans
- » I/O Resource Management
- » Exadata Storage Indexes

### Test Runs and Workload

Each test run was 30 minutes in length which included a restart of the database and a warm up period. OLTP workload was generated using the Order Entry application for Swingbench. A data warehouse reporting application with various queries was used to demonstrate the mixed use case but instance caging was used on the Data Warehouse database in all cases to throttle the Data Warehouse and Reporting queries.

Exalogic servers were used to drive the OLTP application workload. Each Swingbench application had its own separate connection pool.

## Test Results

All test cases incorporated the same MAA consolidation best practices to obtain maximum throughput. For the non-Exadata tests, we continued to add databases until we were resource bound and our average response time remained < 200 ms. We were able to consolidate 40 OLTP databases before we became I/O bound with an average response time of 177 ms. The top wait event was cell single block physical reads (e.g., equivalent to db\_file\_sequential reads) at 47ms with 55% DBtime and log file sync at 122 ms with 31% of DBtime. Average log file parallel write was 23 ms with 20% outliers exceeding 32ms and up to 1 sec. Note, the wait profile and I/O wait times were quite different with just 1 OLTP database or up to 5 OLTP databases. With only a few OLTP databases and without system resource bottlenecks, the database wait profile consisted of 60% DBtime on cell single block physical reads with 4ms average wait time, 30% DBtime with DB CPU and a very low log file sync of 1 ms. With many individual databases competing for IO, CPU and network, waits grow quickly.

After enabling Exadata features, we ran the same tests to consolidate as many individual databases without sacrificing response time or stability. The example below shows the results of consolidating 160 individual OLTP databases on Exadata.

### EXADATA CONSOLIATION TESTING OLTP WORKLOAD: BRONZE TIER

Metrics	Non-Exadata OLTP workload	Exadata OLTP workload	Conclusions
Cumulative TPS	10514	46915	4.4 X improvement
Average response time	177ms	132ms	25% reduction
Number of databases	40	160	4x increase
Top wait events with average wait times	Cell single block physical reads: 47 ms, 55% DBtime Log file sync: 122 ms, 31% DBtime Log file parallel write: 23 ms	Cell single block physical reads: 11 ms; 31% DBtime Log file sync 28 ms, 10% dbtime Log file parallel write: 5ms	Exadata prioritized essential database I/O to ensure low latency. Exadata smart flash extended the IO bandwidth significantly.

Exadata smart flash cache features and network resource management prioritized database read and write operations, especially log write operations, to ensure low latency and high bandwidth. Even with 160 databases, the average log file parallel write was still 5 ms, log file sync at 28 ms and cell single block physical reads were at 11ms. In this example, the workload was CPU bound and the system was very stable.

Consolidation density was successfully increased to 200 databases, 5x the density of the non-Exadata configuration with only an 11% increase in response time as shown below.

INCREASED CONSOLIDATION DENSITY FOR BRONZE TIER

Metrics	Non-Exadata OLTP workload	Exadata OLTP workload	Conclusions
Cumulative TPS	10514	48309	4.5 X improvement
Average response time	177ms	196ms	11% increase
Number of databases	40	200	5x increase
Top wait events with average wait times	Cell single block physical reads: 47 ms, 55% DBtime Log file sync: 122 ms, 31% DBtime Log file parallel write: 23 ms	Cell single block physical reads: 4 ms; 11% DBtime Log file sync 31 ms, 10% dbtime Log file parallel write: 6ms 45% DB time on DBWR type waits such as write complete waits, free buffer waits and buffer busy waits but ran out of time to tune.	Exadata prioritized essential database I/O to ensure low latency. Exadata smart flash extended the IO bandwidth significantly.

Both examples show very good **database consolidation density of 4X to 5X with a pure OLTP workload** for the Bronze tier.

In real world, OLTP applications have some mix of reporting, batch and ETL activities. In a large consolidated environment with 40 or more individual databases and applications, it's common to find non-OLTP like activities. These other activities can have adverse impact on OLTP response time and throughput. In this example, one DW was added for a reporting workload that contained various long running queries. Instance caging was used for the Data Warehouse and reporting workload in all runs to prevent this workload from overwhelming the available I/O and CPU resources. For non-Exadata case, we maxed out the IOPS with 40 OLTP databases plus 1 Data Warehouse/reporting database and for the Exadata case, we maxed the CPU with 160 OLTP databases plus 1 Data Warehouse/reporting database. Once again, Exadata features prioritized key database operations and allowed (6.3 X) higher throughput, significant reduction in response time (50% reduction) and great (4 X) consolidation density.

EXADATA CONSOLIDATION TESTING OLTP +DATA WAREHOUSE AND REPORTING: BRONZE TIER

Metrics	Non-Exadata OLTP workload	Exadata OLTP workload	Conclusions
Cumulative TPS	8466	53662	6.3 X improvement
Average response time	212ms	104ms	50% reduction
Number of databases	41	161	4x increase
Top wait events with average wait times	Cell single block physical reads: 54 ms, 59% DBtime Log file sync: 104 ms, 23% DBtime Log file parallel write: 21 ms	Cell single block physical reads: 4 ms; 13% DBtime Log file sync 38 ms, 21% dbtime Log file parallel write: 4ms	Exadata absorbed the OLTP I/Os and offloaded scans more efficiently

For Silver, Gold, and Platinum applications, Oracle recommends leaving CPU and memory headroom. In this example, CPU and memory were not maxed out on Exadata. The consolidation density is lower, but the reduction in response time and throughput multiplier is much more dramatic.

EXADATA CONSOLIATION TESTING OLTP +DATA WAREHOUSE AND REORTING: SILVER, GOLD AND PLATINUM TIERS

Metrics	Non-Exadata OLTP workload	Exadata OLTP workload	Conclustions
Cumulative TPS	8466	55422	6.5 X improvement
Average response time	212ms	14ms	93% reduction
Number of databases	41	81	2x increase
Top wait events with average wait times	Cell single block physical reads: 54 ms, 59% DBtime Log file sync: 104 ms, 23% DBtime Log file parallel write: 21 ms	DB CPU is top wait at 42% DBtime Cell single block physical reads: 1 ms; 23% DBtime Log file sync 3 ms, 12% dbtime Log file parallel write: 1ms	When not CPU bound, greater performance gains and predictable response times. 2 X Consolidation Density in this example for SILVER+ tiers

To maintain the stability of the environment, Exadata consolidation best practices were applied. For the most part, these configuration best practices are inherent and already applied in our engineered systems. However, the following changes and customizations had the biggest impact:

- » Set HugePages (needs to be adjusted to accommodate all databases).
- » Minimize process count and use connection pools.
- » Set ASM process count.
- » Adjust semmsl and semmns settings.
- » Instance cage the Data Warehouse workload.
- » Use iorm objective=AUTO

## Appendix B: RMAN Active Database Duplication Migration

This appendix provides a high-level overview of the steps used to take a minimal downtime approach for migrating an Oracle Database 12c non-CDB to a PDB. This process takes advantage of the new 12c RMAN Duplicate Active Database functionality using FROM SERVICE to perform an active database backup and perform incremental applies across the network, all without needing additional space for staging any backup files. The files are copied using Oracle Net and stored in the target location of the new CDB. This example assumes ASM is in use for storing the datafiles. At this time, it is only possible to perform a non-CDB to PDB plugin when the source and destination environments are the same-endian. Cross-endian migration requires either Oracle GoldenGate or some method of export/import (traditional export/import, Cross Endian Transportable Tablespace or Cross Endian Full Database Transportable Tablespace).

The following procedure highlights the main steps for plugging in an Oracle Database 12c non-CDB as a PDB into a CDB. A detailed example of the following process along with additional migration methods are defined in [My Oracle Support Note 1576755.1: Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage](#).

To plug in an Oracle Database 12c non-CDB database as a PDB:





- » Upgrade the database to Oracle Database 12c. This database will be a non-CDB.
- » Create a CDB, or choose an existing CDB to be the target of the migration. The CDB creation can be done using either DBCA or with the CREATE DATABASE command via SQL\*Plus. Note that there is no need to precreate a PDB, the PDB will be created during the plugin operation.
- » Optionally, create a physical standby database.
- » Using Oracle Database 12c RMAN, connect to both your original non-CDB and the new CDB. These connections must both use Oracle Net, and the non-CDB should be connected as target and the CDB as a clone database.  

```
RMAN> connect target <user>@non-cdb as sysbkup  
RMAN> connect clone <user>@cdb as sysbkup
```
- » Do the initial active duplicate to copy the image files of the source database to the destination sites. If using OMF, RMAN will restore the files to the correct location.
- » Create a temporary instance on the destination CDB.
- » Restore the non-CDB controlfile to a temporary location on the destination environment.
- » Catalog the just-restored datafiles into the controlfiles on the temporary instance and issue the SWITCH DATAFILE TO NEW command.
- » Create a plugin SQL statement to be used to plug in the non-CDB as a PDB. Use the SOURCE\_FILE\_DIRECTORY clause in the create statement and specify the directory location where the files were restored in step 5.
- » As many times as is necessary, recover using the FROM SERVICE clause, which will perform an incremental backup of the non-CDB and apply the changes to the files on the destination CDB.
- » When you are ready to perform the unplug operation do the following steps:
  - » Close the non-CDB cleanly and open in READ ONLY.
  - » Create the manifest XML file on the non-CDB.
  - » Perform a final incremental apply from the non-CDB to the destination copies.
  - » Copy the manifest to the destination host.
  - » Create the PDB using the manifest XML file.
  - » Open the new PDB.
  - » Backup the new PDB.
  - » Setup application access for the PDB.





CONNECT WITH US

-  [blogs.oracle.com/oracle](https://blogs.oracle.com/oracle)
-  [facebook.com/oracle](https://facebook.com/oracle)
-  [twitter.com/oracle](https://twitter.com/oracle)
-  [oracle.com](https://oracle.com)

**Oracle Corporation, World Headquarters**

500 Oracle Parkway  
Redwood Shores, CA 94065, USA

**Worldwide Inquiries**

Phone: +1.650.506.7000  
Fax: +1.650.506.7200

**Hardware and Software, Engineered to Work Together**

Copyright © 2016, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0416

 | Oracle is committed to developing practices and products that help protect the environment

## ÜBER HUNKLER

Das Karlsruher Systemhaus HUNKLER wurde 1988 erster offizieller Partner von Oracle in Deutschland. Ein Team von rund 20 Mitarbeitern unterstützt Kunden aus Industrie, öffentlicher Verwaltung, Gesundheits- und Finanzwesen mit Beratung, Lösungsentwicklung und Managed Services.

Im Fokus von HUNKLER stehen leistungsfähige, wirtschaftliche Infrastrukturen für Oracle-datenbanken mit den Schwerpunkten Hochverfügbarkeit, Ausfallsicherheit und Zero Downtime Migration. Die integrierten Komplettlösungen der Produktfamilie Oracle Engineered Systems sowie der Datenbank-/Anwendungsbetrieb in der Oracle Cloud sind weitere Themenfelder, die das Unternehmen umfassend abdeckt.

**HUNKLER**  
GmbH & Co. KG

### **Hauptsitz Karlsruhe**

Bannwaldallee 32, 76185 Karlsruhe  
Tel. 0721-490 16-0, Fax 0721-490 16-29

### **Geschäftsstelle Bodensee**

Fritz-Reichle-Ring 6a  
78315 Radolfzell  
Tel. 07732-939 14-00, Fax 07732-939 14-04  
**info@hunkler.de, www.hunkler.de**