Oracle® Database

High Availability Overview and Best Practices

F23691-05
November 2021

Oracle Database High Availability Overview and Best Practices,

F23691-05

Contents

Preface

Part I Oracle Database High Availability Overview

1 Overview of High Availability

2 High Availability and Data Protection – Getting From Requirements to Architecture

Part II Oracle Database High Availability Best Practices

8 Overview of Oracle Database High Availability Best Practices

9 Oracle Database Configuration Best Practices

Part III Oracle Data Guard Best Practices

10 Overview of MAA Best Practices for Oracle Data Guard

11 Plan an Oracle Data Guard Deployment

12 Configure and Deploy Oracle Data Guard

13 Tune and Troubleshoot Oracle Data Guard

14 Monitor an Oracle Data Guard Configuration

Preface

This book introduces you to Oracle best practices for deploying a highly available database environment, and provides best practices for configuring the Oracle MAA reference architectures.

Part 1 provides an overview of high availability and helps you to determine your high availability requirements. It describes the Oracle Database products and features that are designed to support high availability and describes the primary database architectures that can help your business achieve high availability.

Part 2 describes the best practices for configuring a highly available Oracle database, using features provided with Oracle Database, which lets you achieve MAA Bronze reference architecture service levels

Part 3 describes the best practices for configuring a highly available Oracle database using Oracle Data Guard for replication and data protection, which lets you achieve MAA Gold reference architecture service levels.

This preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This book is intended for chief technology officers, information technology architects, database administrators, system administrators, network administrators, and application administrators who perform the following tasks:

- Plan data centers
- Implement data center policies
- Maintain high availability systems
- Plan and build high availability solutions

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

Knowledge of Oracle Database, Oracle RAC, and Data Guard concepts and terminology is required to understand the configuration and implementation details described in this book. For more information, see the Oracle Database documentation set. These books may be of particular interest:

- *Oracle Database Administrator’s Guide*
- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle Real Application Clusters Administration and Deployment Guide*
- *Oracle Automatic Storage Management Administrator's Guide*
- *Oracle Data Guard Concepts and Administration*
- *Oracle Database Backup and Recovery User's Guide*

Many books in the documentation set use the sample schemas of the seed database, which is installed by default when you install Oracle Database. See *Oracle Database Sample Schemas* for information about using these schemas.

Also, you can download the Oracle MAA best practice white papers at http://www.oracle.com/goto/maa.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I

Oracle Database High Availability Overview

1
Overview of High Availability

See the following topics to learn what high availability and why it is important. Then follow the roadmap to implementing a Maximum Availability Architecture.

What Is High Availability?

Availability is the degree to which an application and database service is available.

Availability is measured by the perception of an application's user. Users experience frustration when their data is unavailable or the computing system is not performing as expected, and they do not understand or care to differentiate between the complex components of an overall solution. Performance failures due to higher than expected usage create the same disruption as the failure of critical components in the architecture. If a user cannot access the application or database service, it is said to be unavailable. Generally, the term downtime is used to refer to periods when a system is unavailable.

Users who want their systems to be always ready to serve them need high availability. A system that is highly available is designed to provide uninterrupted computing services during essential time periods, during most hours of the day, and most days of the week throughout the year; this measurement is often shown as 24x365. Such systems may also need a high availability solution for planned maintenance operations such as upgrading a system's hardware or software.

Reliability, recoverability, timely error detection, and continuous operations are primary characteristics of a highly available solution:

- Reliability: Reliable hardware is one component of a high availability solution. Reliable software—including the database, web servers, and applications—is just as critical to implementing a highly available solution. A related characteristic is resilience. For example, low-cost commodity hardware, combined with software such as Oracle Real Application Clusters (Oracle RAC), can be used to implement a very reliable system. The resilience of an Oracle RAC database allows processing to continue even though individual servers may fail. For example, the Oracle RAC database allows processing to continue even though individual servers may fail.
- Recoverability: Even though there may be many ways to recover from a failure, it is important to determine what types of failures may occur in your high availability environment and how to recover from those failures quickly in order to meet your business requirements. For example, if a critical table is accidentally deleted from the database, what action should you take to recover it? Does your architecture provide the ability to recover in the time specified in a service-level agreement (SLA)?
- Timely error detection: If a component in your architecture fails, then fast detection is essential to recover from the unexpected failure. Although you may be able to recover quickly from an outage, if it takes an additional 90 minutes to discover the problem, then you may not meet your SLA. Monitoring the health of your environment requires reliable software to view it quickly and the ability to notify the database administrator of a problem.
- Continuous operation: Providing continuous access to your data is essential when very little or no downtime is acceptable to perform maintenance activities. Activities, such as

moving a table to another location in the database or even adding CPUs to your hardware, should be transparent to the user in a high availability architecture.

More specifically, a high availability architecture should have the following traits:

- Tolerate failures such that processing continues with minimal or no interruption
- Be transparent to—or tolerant of—system, data, or application changes
- Provide built-in preventive measures
- Provide active monitoring and fast detection of failures
- Provide fast recoverability
- Automate detection and recovery operations
- Protect the data to minimize or prevent data loss and corruptions
- Implement the operational best practices to manage your environment
- Achieve the goals set in SLAs (for example, recovery time objectives (RTOs) and recovery point objectives (RPOs)) for the lowest possible total cost of ownership

Importance of Availability

The importance of high availability varies among applications. Databases and the internet have enabled worldwide collaboration and information sharing by extending the reach of database applications throughout organizations and communities.

This reach emphasizes the importance of high availability in data management solutions. Both small businesses and global enterprises have users all over the world who require access to data 24 hours a day. Without this data access, operations can stop, and revenue is lost. Users now demand service-level agreements from their information technology (IT) departments and solution providers, reflecting the increasing dependence on these solutions. Increasingly, availability is measured in dollars, euros, and yen, not just in time and convenience.

Enterprises have used their IT infrastructure to provide a competitive advantage, increase productivity, and empower users to make faster and more informed decisions. However, with these benefits has come an increasing dependence on that infrastructure. If a critical application becomes unavailable, then the business can be in jeopardy. The business might lose revenue, incur penalties, and receive bad publicity that has a lasting effect on customers and on the company's stock price.

It is important to examine the factors that determine how your data is protected and maximize availability to your users.

Cost of Downtime

The need to deliver increasing levels of availability continues to accelerate as enterprises reengineer their solutions to gain competitive advantage. Most often, these new solutions rely on immediate access to critical business data.

When data is not available, the operation can cease to function. Downtime can lead to lost productivity, lost revenue, damaged customer relationships, bad publicity, and lawsuits.

It is not always easy to place a direct cost on downtime. Angry customers, idle employees, and bad publicity are all costly, but not directly measured in currency. On the other hand, lost revenue and legal penalties incurred because SLA objectives are not met can easily be quantified. The cost of downtime can quickly grow in industries that are dependent on their solutions to provide service.

Other factors to consider in the cost of downtime are:

- The maximum tolerable length of a single unplanned outage

  If the event lasts less than 30 seconds, then it may cause very little impact and may be barely perceptible to users. As the length of the outage grows, the effect may grow exponentially and negatively affect the business.

- The maximum frequency of allowable incidents

  Frequent outages, even if short in duration, may similarly disrupt business operations.

When designing a solution, it is important to recognize the true cost of downtime to understand how the business can benefit from availability improvements.

Oracle provides a range of high availability solutions to fit every organization regardless of size. Small workgroups and global enterprises alike are able to extend the reach of their critical business applications. With Oracle and the Internet, applications and data are reliably accessible everywhere, at any time.

Causes of Downtime

One of the challenges in designing a high availability solution is examining and addressing all of the possible causes of downtime.

It is important to consider causes of both unplanned and planned downtime when designing a fault-tolerant and resilient IT infrastructure. Planned downtime can be just as disruptive to operations as unplanned downtime, especially in global enterprises that support users in multiple time zones.

The following table describes unplanned outage types and provides examples of each type.

Table 1-1 Causes of Unplanned Downtime

Type	Description	Examples
Site failure	A site failure may affect all processing at a data center, or a subset of applications supported by a data center. The definition of site varies given the contexts of on-premises and cloud. • Site failure - entire regional failure • Data center - entire data center location • Availability domain - isolated data center within a region with possibly many other availability domains • Fault domain - isolated set of system resources within an Availability Domain or data center Typically, each site, data center, availability domain, and fault domain has its own set of isolated hardware, DB compute, network, storage, and power.	• Extended site-wide power failure • Site-wide network failure • Natural disaster makes a data center inoperable • Terrorist or malicious attack on operations or the site

Table 1-1 (Cont.) Causes of Unplanned Downtime

Type	Description	Examples
Cluster-wide failure	The whole cluster hosting an Oracle RAC database is unavailable or fails. This includes: • Failures of nodes in the cluster • Failure of any other components that result in the cluster being unavailable and the Oracle database and instances on the site being unavailable	• The last surviving node on the Oracle RAC cluster fails and the node or database cannot be restarted • Both redundant cluster interconnections fail or Clusterware failure • Database corruption so severe that continuity is not possible on the current database server • Clusterware and hardware-software defects preventing availability or stability.
Computer failure	A computer failure outage occurs when the system running the database becomes unavailable because it has failed or is no longer available. When the database uses Oracle RAC then a computer failure represents a subset of the system (while retaining full access to the data).	• Database system hardware failure • Operating system failure • Oracle instance failure
Network failure	A network failure outage occurs when a network device stops or reduces network traffic and communication from your application to database, database to storage, or any system to system that is critical to your application service processing.	• Network switch failure • Network interface failure • Network cable failures
Storage failure	A storage failure outage occurs when the storage holding some or all of the database contents becomes unavailable because it has shut down or is no longer available.	• Disk or flash drive failure • Disk controller failure • Storage array failure

Table 1-1 (Cont.) Causes of Unplanned Downtime

Type	Description	Examples
Data corruption	A corrupt block is a block that was changed so that it differs from what Oracle Database expects to find. Block corruptions can be categorized as physical or logical: • In a physical block corruption, which is also called a media corruption, the database does not recognize the block at all; the checksum is invalid or the block contains all zeros. An example of a more sophisticated block corruption is when the block header and footer do not match. • In a logical block corruption, the contents of the block are physically sound and pass the physical block checks; however, the block can be logically inconsistent. Examples of logical block corruption include incorrect block type, incorrect data or redo block sequence number, corruption of a row piece or index entry, or data dictionary corruptions. Block corruptions can also be divided into interblock corruption and intrablock corruption: • In an intrablock corruption, the corruption occurs in the block itself and can be either a physical or a logical block corruption. • In an interblock corruption, the corruption occurs between blocks and can only be a logical block corruption. A data corruption outage occurs when a hardware, software, or network component causes corrupt data to be read or written. The service-level impact of a data corruption outage may vary, from a small portion of the application or database (down to a single database block) to a large portion of the application or database (making it essentially unusable).	• Operating system or storage device driver failure • Faulty host bus adapter • Disk controller failure • Volume manager error causing a bad disk read or write • Software or hardware defects
Human error	A human error outage occurs when unintentional or other actions are committed that cause data in the database to become incorrect or unusable. The service-level impact of a human error outage can vary significantly, depending on the amount and critical nature of the affected data.	• File deletion (at the file system level) • Dropped database object • Inadvertent data changes • Malicious data changes

Table 1-1 (Cont.) Causes of Unplanned Downtime

Type	Description	Examples
Lost or stray writes	A lost or stray write is another form of data corruption, but it is much more difficult to detect and repair quickly. A data block stray or lost write occurs when: • For a lost write, an I/O subsystem acknowledges the completion of the block write even though the write I/O did not occur in the persistent storage. On a subsequent block read on the primary database, the I/O subsystem returns the stale version of the data block, which might be used to update other blocks of the database, thereby corrupting it. • For a stray write, the write I/O completed but it was written somewhere else, and a subsequent read operation returns the stale value. • For an Oracle RAC system, a read I/O from one cluster node returns stale data after a write I/O is completed from another node (lost write). For example, this occurs if a network file system (NFS) is mounted in Oracle RAC without disabling attribute caching (for example, without using the `noac` option). In this case, the write I/O from one node is not immediately visible to another node because it is cached. Block corruptions caused by stray writes or lost writes can cause havoc to your database availability. The data block may be physically or logically correct but subsequent disk reads will show blocks that are stale or with an incorrect Oracle Database block address.	• Operating system or storage device driver failure • Faulty host bus adapter • Disk controller failure • Volume manager error • Other application software • Lack of network file systems (NFS) write visibility across a cluster • Software or hardware defects
Delay, slowdown, or hangs	A delay or slowdown occurs when the database or the application cannot process transactions because of a resource or lock contention. A perceived delay can be caused by lack of system resources.	• Database or application deadlocks • Runaway processes that consume system resources • Logon storms or system faults • Combination of application peaks with lack of system or database resources. This can occur with one application or many applications in a consolidated database environment without proper resource management. • Archived redo log destination or fast recovery area destination becomes full • Oversubscribed or heavily consolidated database system

The following table describes planned outage types and provides examples of each type.

Table 1-2 Causes of Planned Downtime

Type	Description	Examples
Software changes	• Planned periodic software changes to apply minor fixes for stability and security • Planned annual or bi-annual major upgrades to adopt new features and capabilities	• Software updates, including security updates to operating system, clusterware. or database • Major upgrade of operating system, clusterware, or database • Updating or upgrading application software
System and database changes	• Planned system changes to replace defected hardware • Planned system changes to expand or reduce system resources • Planned database changes to adopt parameter changes • Planned change to migrate to new hardware or architecture	• Adding or removing processors or memory to a server • Adding or removing nodes to or from a cluster • Adding or removing disks drives or storage arrays • Replacing any Field Replaceable Unit (FRU) • Changing configuration parameters • System platform migration • Migrating to cluster architecture • Migrating to new storage
Data changes	Planned data changes to the logical structure or physical organization of Oracle Database objects. The primary objective of these changes is to improve performance or manageability.	• Table definition changes • Adding table partitioning • Creating and rebuilding indexes
Application changes	Planned application changes can include data changes and schema and programmatic changes. The primary objective of these changes is to improve performance, manageability, and functionality.	Application upgrades

Oracle offers high availability solutions to help avoid both unplanned and planned downtime, and recover from failures. Oracle Database High Availability Solutions for Unplanned Downtime and Oracle Database High Availability Solutions for Planned Downtime discuss each of these high availability solutions in detail.

Chaos Engineering

Maximum Availability Architecture leverages Chaos Engineering throughout its testing and development life cycles to ensure that end-to-end application and database availability is preserved or at its optimal levels for any fault or maintenance event.

Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system's capability to withstand turbulent conditions in production. Specifically, MAA injects various faults and planned maintenance events to evaluate application and database impact throughout our development, stress, and testing cycles. With that experimentation, best practices, defects, and lessons learned are derived, and that knowledge is put back in practice to evolve and improve our MAA solutions.

Roadmap to Implementing the Maximum Availability Architecture

Oracle high availability solutions and sound operational practices are the key to successful implementation of an IT infrastructure. However, technology alone is not enough.

Choosing and implementing an architecture that best fits your availability requirements can be a daunting task. Oracle Maximum Availability Architecture (MAA) simplifies the process of choosing and implementing a high availability architecture to fit your business requirements with the following considerations:

- Encompasses redundancy across all components
- Provides protection and tolerance from computer failures, storage failures, human errors, data corruption, lost writes, system delays or slowdowns, and site disasters
- Recovers from outages as quickly and transparently as possible
- Provides solutions to eliminate or reduce planned downtime
- Provides consistent high performance and robust security
- Provides Oracle Engineered System and cloud options to simplify deployment and management and achieve the highest scalability, performance, and availability
- Achieves SLAs at the lowest possible total cost of ownership
- Applies to On-Premise, Oracle Public Cloud, and hybrid architectures consisting of parts on-premise and part in the cloud
- Provides special consideration to Container or Oracle Multitenant, Oracle Database In-Memory, and Oracle Sharding architectures

To build, implement, and maintain this type of architecture, you need to:

1. Analyze your specific high availability requirements, including both the technical and operational aspects of your IT systems and business processes, as described in High Availability and Data Protection – Getting From Requirements to Architecture.
2. Evaluate the various high availability architectures and their benefits and options, as described in Oracle MAA Reference Architectures.
3. Understand the availability impact for each MAA reference architecture, or various high availability features, on businesses and applications, as described in Oracle Database High Availability Solutions for Unplanned Downtime, and Oracle Database High Availability Solutions for Planned Downtime.
4. Familiarize yourself with Oracle high availability features, as described in Features for Maximizing Availability.
5. Use operational best practices to provide a successful MAA implementation, as described in Operational Prerequisites to Maximizing Availability .
6. Implement a high availability architecture using Oracle MAA resources, which provide technical details about the various Oracle MAA high availability technologies, along with best practice recommendations for configuring and using such technologies, such as Oracle MAA best practices white papers, customer

papers with proof of concepts, customer case studies, recorded web casts, demonstrations, and presentations.

Additional Oracle MAA resources are available at http://www.oracle.com/goto/maa.

2

High Availability and Data Protection – Getting From Requirements to Architecture

See the following topics to learn how Oracle Maximum Availability Architecture provides a framework to effectively evaluate the high availability requirements of an enterprise.

High Availability Requirements

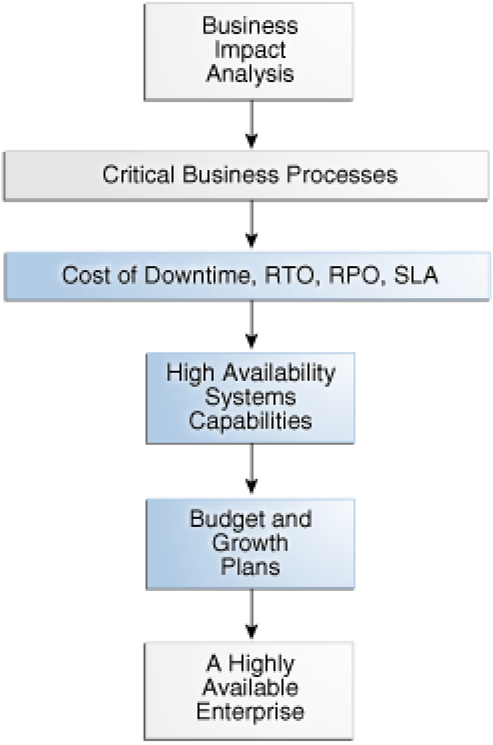
Any effort to design and implement a high availability strategy for Oracle Database begins by performing a thorough business impact analysis to identify the consequences to the enterprise of downtime and data loss, whether caused by unplanned or planned outages.

The term "business impact" is intended to be agnostic of whether the enterprise is a commercial venture, government agency, or not-for-profit institution. In all cases, data loss and downtime can seriously impact the ability of any enterprise to perform its functions. Implementing high availability may involve critical tasks such as:

- Retiring legacy systems
- Investing in more capable and robust systems and facilities
- Redesigning the overall IT architecture and operations to adapt to this high availability model
- Modifying existing applications to take full advantage of high availability infrastructures
- Redesigning business processes
- Hiring and training personnel
- Moving parts or an entire application or database into the Oracle Public Cloud
- Balancing the right level of consolidation, flexibility, and isolation
- Understanding the capabilities and limitations of your existing system and network infrastructure

By combining your business analysis with an understanding of the level of investment required to implement different high availability solutions, you can develop a high availability architecture that achieves both business and technical objectives.

Figure 2-1 Planning and Implementing a Highly Available Enterprise


Business Impact Analysis
Critical Business Processes
Cost of Downtime, RTO, RPO, SLA
High Availability Systems Capabilities
Budget and Growth Plans
A Highly Available Enterprise

A Methodology for Documenting High Availability Requirements

The elements of this analysis framework are:

- Business Impact Analysis
- Cost of Downtime
- Recovery Time Objective
- Recovery Point Objective
- Manageability Goal
- Total Cost of Ownership and Return on Investment

Business Impact Analysis

The business impact analysis categorizes the business processes based on the severity of the impact of IT-related outages.

A rigorous business impact analysis:

- Identifies the critical business processes in an organization
- Calculates the quantifiable loss risk for unplanned and planned IT outages affecting each of these business processes

- Outlines the effects of these outages
- Considers essential business functions, people and system resources, government regulations, and internal and external business dependencies
- Is based on objective and subjective data gathered from interviews with knowledgeable and experienced personnel
- Reviews business practice histories, financial reports, IT systems logs, and so on

For example, consider a semiconductor manufacturer with chip fabrication plants located worldwide. Semiconductor manufacturing is an intensely competitive business requiring a huge financial investment that is amortized over high production volumes. The human resource applications used by plant administration are unlikely to be considered as mission-critical as the applications that control the manufacturing process in the plant. Failure of the applications that support manufacturing affects production levels and have a direct impact on the financial results of the company.

As another example, an internal knowledge management system is likely to be considered mission-critical for a management consulting firm, because the business of a client-focused company is based on internal research accessibility for its consultants and knowledge workers. The cost of downtime of such a system is extremely high for this business.

Similarly, an e-commerce company is highly dependent on customer traffic to its website to generate revenue. Any disruption in service and loss of availability can dampen customer experience and drive away customers to the competition. Thus, the company needs to ensure that the existing infrastructure can scale and handle spikes in customer traffic. Sometimes, this is not possible using on-premise hardware and by moving the cloud the company can ensure their systems always remain operational.

Cost of Downtime

A complete business impact analysis provides the insight needed to quantify the cost of unplanned and planned downtime.

Understanding this cost is essential because it helps prioritize your high availability investment and directly influences the high availability technologies that you choose to minimize the downtime risk.

Various reports have been published, documenting the costs of downtime in different industries. Examples include costs that range from millions of dollars for each hour of brokerage operations and credit card sales, to tens of thousands of dollars for each hour of package shipping services.

These numbers are staggering. The Internet and Cloud can connect the business directly to millions of customers. Application downtime can disrupt this connection, cutting off a business from its customers. In addition to lost revenue, downtime can negatively affect customer relationships, competitive advantages, legal obligations, industry reputation, and shareholder confidence.

Recovery Time Objective

The business impact analysis determines your tolerance to downtime, also known as the recovery time objective (RTO).

An RTO is defined as the maximum amount of time that an IT-based business process can be down before the organization starts suffering unacceptable consequences (financial losses, customer dissatisfaction, reputation, and so on). RTO indicates the downtime tolerance of a business process or an organization in general.

RTO requirements are driven by the mission-critical nature of the business. Therefore, for a system running a stock exchange, the RTO is zero or near to zero.

An organization is likely to have varying RTO requirements across its various business processes. A high volume e-commerce website, for which there is an expectation of rapid response times, and for which customer switching costs are very low, the web-based customer interaction system that drives e-commerce sales is likely to have an RTO of zero or close to zero. However, the RTO of the systems that support back-end operations, such as shipping and billing, can be higher. If these back-end systems are down, then the business may resort to manual operations temporarily without a significant visible impact.

Some organizations have varying RTOs based on the probability of failures. One simple class separation is local failures (such as single database compute, disk/flash, network failure) as opposed to disasters (such as a complete cluster, database, data corruptions, or a site failure). Typically, business-critical customers have an RTO of less than 1 minute for local failures, and may have a higher RTO of less than 1 hour for disasters. For mission-critical applications the RTOs may indeed be the same for all unplanned outages.

Recovery Point Objective

The business impact analysis also determines your tolerance to data loss, also known as a recovery point objective (RPO).

The RPO is the maximum amount of data that an IT-based business process can lose without harm to the organization. RPO measures the data-loss tolerance of a business process or an organization in general. This data loss is often measured in terms of time, for example, zero, seconds, hours, or days of data loss.

A stock exchange where millions of dollars worth of transactions occur every minute cannot afford to lose any data. Therefore, its RPO must be zero. The web-based sales system in the e-commerce example does not require an RPO of zero, although a low RPO is essential for customer satisfaction. However, its back-end merchandising and inventory update system can have a higher RPO because lost data can be reentered.

An RPO of zero can be challenging for disasters, but I can be accomplished with various Oracle technologies protecting your database, especially Zero Data Loss Recovery Appliance.

Manageability Goal

A manageability goal is more subjective than either the RPO or the RTO. You must make an objective evaluation of the skill sets, management resources, and tools available in an organization, and the degree to which the organization can successfully manage all elements of a high availability architecture.

Just as RPO and RTO measure an organization's tolerance for downtime and data loss, your manageability goal measures the organization's tolerance for complexity in the IT environment. When less complexity is a requirement, simpler methods of achieving high availability are preferred over methods that may be more complex to manage, even if the latter could attain more aggressive RTO and RPO objectives.

Understanding manageability goals helps organizations differentiate between what is possible and what is practical to implement.

Moving Oracle databases to Oracle Cloud can reduce manageability cost and complexity significantly, because Oracle Cloud lets you to choose between various Maximum Availability Architecture architectures with built-in configuration and life cycle operations. With Autonomous Database Cloud, database life cycle operations, such as backup and restore, software updates, and key repair operations are automatic.

Total Cost of Ownership and Return on Investment

Understanding the total cost of ownership (TCO) and objectives for return on investment (ROI) are essential to selecting a high availability architecture that also achieves the business goals of your organization.

TCO includes all costs (such as acquisition, implementation, systems, networks, facilities, staff, training, and support) over the useful life of your chosen high availability solution. Likewise, the ROI calculation captures all of the financial benefits that accrue for a given high availability architecture.

For example, consider a high availability architecture in which IT systems and storage at a remote standby site remain idle, with no other business use that can be served by the standby systems. The only return on investment for the standby site is the costs related to downtime avoided by its use in a failover scenario. Contrast this with a different high availability architecture that enables IT systems and storage at the standby site to be used productively while in the standby role (for example, for reports or for off-loading the overhead of user queries or distributing read-write workload from the primary system). The return on investment of such an architecture includes both the cost of downtime avoided and the financial benefits that accrue to its productive use, while also providing high availability and data protection.

Enterprises can also reduce TCO for growing infrastructure needs by moving workloads to the cloud rather than making an upfront capital investment in building a new data center. The major economic appeal is to convert capital expenditures into operational expenditures, and generate a higher ROI.

Mapping Requirements to Architectures

The business impact analysis will help you document what is already known. The outcome of the business impact analysis provides the insight you need to group databases having similar RTO and RPO objectives together.

Different applications, and the databases that support them, represent varying degrees of importance to the enterprise. A high level of investment in high availability infrastructure may not make sense for an application that if down, would not have an immediate impact on the enterprise. So where do you start?

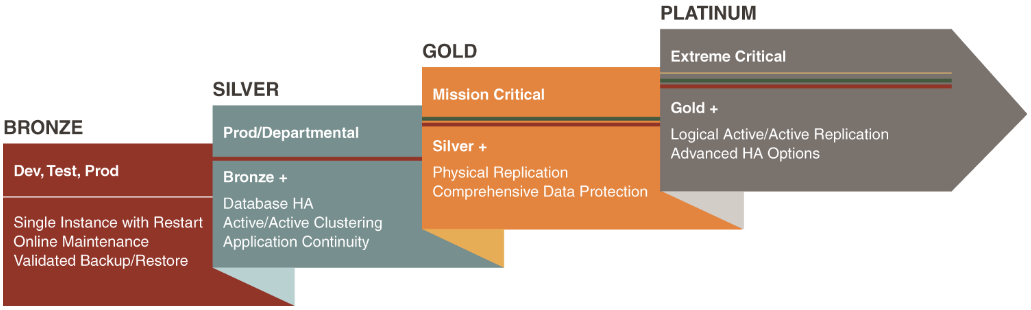
Groups of databases by similar RTO and RPO can be mapped to a controlled set of high availability reference architectures that most closely address the required service levels. Note that in the case where there are dependencies between databases, they are grouped with the database having the most stringent high availability requirement.

Oracle MAA Reference Architectures

Oracle MAA best practices define high availability reference architectures that address the complete range of availability and data protection required by enterprises of all sizes and lines of business.

The Platinum, Gold, Silver, and Bronze MAA reference architectures, or tiers, are applicable to on-premises, private and public cloud configurations, and hybrid cloud. They deliver the service levels described in the following figure.

Figure 2-2 Oracle MAA Reference Architectures



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

Container databases (CDBs) using Oracle Multitenant can exist in any tier, Bronze through Platinum, providing higher consolidation density and higher TCO. Typically, the consolidation density is higher with Bronze and Silver tiers, and there is less or zero consolidation when deploying a Platinum tier.

Oracle Database In-Memory can also be leveraged in any of the MAA tiers. Because the In-Memory column store is seamlessly integrated into Oracle Database, all of the high availability benefits that come from the MAA tiers are inherited when implementing Oracle Database In-Memory.

Oracle Engineered Systems can also exist in any of the tiers. Integrating Zero Data Loss Recovery Appliance (Recovery Appliance) as the Oracle Database backup and recovery solution for your entire data center reduces RPO and RTO when restoring from backups. Leveraging Oracle Exadata Database Machine as your database platform in the MAA reference architectures provides the best database platform solution with the lowest RTO and brownout, along with additional Exadata MAA quality of service.

See Also:

High Availability Reference Architectures

Oracle Exadata Database Machine: Maximum Availability Architecture and MAA Best Practices for Oracle Exadata Database Machine

http://www.oracle.com/goto/maa for MAA white paper “Oracle Database In-Memory High Availability Best Practices”

Bronze Reference Architecture

The Bronze tier is appropriate for databases where simple restart of a failed component (e.g. listener, database instance, or database) or restore from backup is "HA and DR enough."

The Bronze reference architecture is based on a single instance Oracle Database using MAA best practices that implement the many capabilities for data protection and high availability included with every 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 restarting. The Bronze architecture then uses a redundant system infrastructure enhanced by Oracle's technologies, such as Oracle Restart, Recovery Manager (RMAN), Zero Data Loss Recovery Appliance, Flashback technologies, Online Redefinition, Online Patching, Automatic Storage Management (ASM), Oracle Multitenant, and more.

Silver Reference Architecture

The Silver tier provides an additional level of high availability for databases that require minimal or zero downtime in the event of database instance or server failure, as well as most common planned maintenance events, such as hardware and software updates.

The Silver reference architecture adds a rich set of enterprise capabilities and benefits, including clustering technology using either Oracle RAC or Oracle RAC One Node. Also, Application Continuity provides a reliable replay of in-flight transactions, which masks outages from users and simplifies application failover.

Gold Reference Architecture

The Gold tier raises the stakes substantially for business-critical applications that cannot tolerate high RTO and RPO for any disasters such as database, cluster, corruptions, or site failures. Additionally, major database upgrades or site migrations can be done in seconds.

The Gold tier also reduces costs while improving your return on investment by actively using all of the replicas at all times.

The Gold reference architecture adds database-aware replication technologies, Oracle Data Guard and Oracle Active Data Guard, which synchronize one or more replicas of the production database to provide real time data protection and availability. Database-aware replication substantially enhances high availability and data protection (corruption protection) beyond what is possible with storage replication technologies. Oracle Active Data Guard Far Sync is used for zero data loss protection at any distance.

See also, Oracle Data Guard Advantages Over Traditional Solutions.

Platinum Reference Architecture

The Platinum tier introduces several new Oracle Database capabilities, including Oracle GoldenGate for zero-downtime upgrades and migrations.

Edition Based Redefinition lets application developers design for zero-downtime application upgrades. You can alternativly design applications for Oracle Sharding, which provides extreme availability by distributing subsets of a database into highly available shards, while the application can access the entire database as one single logical database.

Each of these technologies requires additional effort to implement, but they deliver substantial value for the most critical applications where downtime is not an option.

High Availability and Data Protection Attributes by Tier

Each MAA reference architecture delivers known and tested levels of downtime and data protection.

The following table summarizes the high availability and data protection attributes inherent to each architecture. Each architecture includes all of the capabilities of the previous architecture, and builds upon it to handle an expanded set of outages. The various components included and the service levels achieved by each architecture are described in other topics.

Table 2-1 High Availability and Data Protection Attributes By MAA Reference Architecture

MAA Reference Architecture	Unplanned Outages (Local Site)	Planned Maintenance	Data Protection	Unrecoverable Local Outages and Disaster Recovery
Bronze	Single Instance, auto-restart for recoverable instance and server failures. Redundancy for system infrastructure so that single component failures such as disk, flash, and network should not result in downtime.	Some online, most off-line	Basic runtime validation combined with manual checks	Restore from backup, potential to lose data generated since the last backup. Using Zero Data Loss Recovery Appliance reduces the potential to lose data to zero or near zero.

Table 2-1 (Cont.) High Availability and Data Protection Attributes By MAA Reference Architecture

MAA Reference Architecture	Unplanned Outages (Local Site)	Planned Maintenance	Data Protection	Unrecoverable Local Outages and Disaster Recovery
Silver	HA with automatic failover for instance and server failures	Most rolling, some online, few offline	Basic runtime validation combined with manual checks	Restore from backup, potential to lose data generated since the last backup. Using Zero Data Loss Recovery Appliance reduces the potential to lose data to zero or near zero. In-flight transactions are preserved with Application Continuity.
Gold	Comprehensive high availability and disaster recovery	All rolling or online	Comprehensive runtime validation combined with manual checks	Real-time failover, zero or near-zero data loss
Platinum	Zero application outage for Platinum ready applications	Zero application outage	Comprehensive runtime validation combined with manual checks	Zero application outage for Platinum-ready applications, with zero data loss. Oracle RAC, Oracle Active Data Guard, and Oracle GoldenGate complement each other, providing a wide array of solutions to achieve zero database service downtime for unplanned outages. Alternatively, use Oracle Sharding for site failure protection, because impact on the application is only on shards in failed site rather than the entire database. Each shard can be configured with real-time failover, zero or near-zero data loss, or zero application outage for Platinum-ready applications. In-flight transactions are preserved, with zero data loss.

See Also:

http://www.oracle.com/goto/maa

3

Features for Maximizing Availability

Familiarize yourself with the following Oracle Database high availability features used in MAA solutions.

Oracle Data Guard

Oracle Data Guard ensures high availability, data protection, and disaster recovery for enterprise data.

Data Guard provides a comprehensive set of services that create, maintain, manage, and monitor one or more standby databases to enable Oracle databases to survive outages of any kind, including natural disasters and data corruptions. A Data Guard standby database is an exact replica of the production database and thus can be transparently utilized in combination with traditional backup, restoration, flashback, and cluster techniques to provide the highest possible level of data protection, data availability and disaster recovery. Data Guard is included in Oracle Enterprise Edition.

A Data Guard configuration consists of one primary database and one or more standby databases. A primary database can be either a single-instance Oracle database or an Oracle RAC database. Similar to a primary database, a standby database can be either a single-instance Oracle database or an Oracle RAC database. Using a backup copy of the primary database, you can create up to 30 standby databases that receive redo directly from the primary database. Optionally you can use a cascaded standby to create Data Guard configurations where the primary transmits redo to a single remote destination, and that destination forwards redo to multiple standby databases. This enables a primary database to efficiently synchronize many more than 30 standby databases if desired.

Note:

Oracle Active Data Guard is an extension of basic Data Guard providing advanced features that off-load various types of processing from a production database, extend zero data loss protection over any distance, and that enhance high availability. Oracle Active Data Guard is licensed separately from Oracle Database Enterprise Edition.

There are several types of standby databases. Data Guard physical standby database is the MAA best practice for data protection and disaster recovery and is the most common type of standby database used. A physical standby database uses Redo Apply (an extension of Oracle media recovery) to maintain an exact, physical replica of the production database. When configured using MAA best practices, Redo Apply uses multiple Oracle-aware validation checks to prevent corruptions that can impact a primary database from impacting the standby. Other types of Data Guard standby databases include: snapshot standby (a standby open read/write for test or other purposes) and logical standby (used to reduce planned downtime).

Benefits of Using Data Guard

- Continuous Oracle-aware validation of all changes using multiple checks for physical and logical consistency of structures within an Oracle data block and redo, before updates are applied to a standby database. This isolates the standby database and prevents it from being impacted by data corruptions that can occur on the primary system.

- Transparent operation: There are no restrictions on the use of Data Guard physical standby for data protection. Redo Apply supports all data and storage types, all DDL operations, and all applications (custom and packaged applications), and guarantees data consistency across primary and standby databases.
- Highest performance: Fast redo transport for best recovery point objective, fast apply performance for best recovery time objective. Multi-instance redo apply provides Oracle RAC scalability for redo apply, eliminating bottlenecks of a single database server. Redo apply can essentially scale up to available CPU, I/O, and network across your Oracle RAC cluster. An observed redo apply rate of 3500 MB per second (12 TB/hour) on 8 node RAC Exadata.
- Fast failover to a standby database to maintain availability should the primary database fail for any reason. Failover is either a manual or automatic operation depending on how Data Guard is configured.
- Integrated client notification framework to enable application clients to connect to a new primary database after a failover occurs.
- Automatic or automated (depending upon configuration) resynchronization of a failed primary database, quickly converting it to a synchronized standby database after a failover occurs.
- Choice of flexible data protection levels to support all network configurations, availability and performance SLAs, and business requirements.
- Management of a primary and all of its standby databases as a single configuration to simplify management and monitoring using either the Data Guard Broker command-line interface or Oracle Enterprise Manager Cloud Control.
- Data Guard Broker greatly improves manageability with additional features for comprehensive configuration health checks, resumable switchover operations, streamlined role transitions, support for cascaded standby configurations, and user-configurable thresholds for transport and apply lag to automatically monitor the ability of the configuration to support SLAs for recovery point and recovery time objectives at any instant in time.
- Efficient transport to multiple remote destinations using a single redo stream originating from the primary production database and forwarded by a cascading standby database.
- Snapshot Standby enables a physical standby database to be open read/write for testing or any activity that requires a read/write replica of production data. A snapshot standby continues to receive but does not apply updates generated by the primary. When testing is complete, a snapshot standby is converted back into a synchronized physical standby database by first discarding the changes made during the open read/write, and then applying the redo received from the primary database. Primary data is always protected. Snapshot standby is particularly useful when used in conjunction with Oracle Real Application Testing (workload is captured at the production database for replay and subsequent performance analysis at the standby database-an exact replica of production).
- Reduction of planned downtime by using a standby database to perform maintenance in a rolling manner. The only downtime is the time required to perform a Data Guard switchover; applications remain available while the maintenance is being performed.
- Increased flexibility for Data Guard configurations where the primary and standby systems may have different CPU architectures or operating systems subject to limitations defined in My Oracle Support note 413484.1.

- Efficient disaster recovery for a container database (CDB). Data Guard failover and switchover completes using a single command at a CDB level regardless of how many pluggable databases (PDBs) are consolidated within the CDB.
- Enables a specific administration privilege, SYSDG, to handle standard administration duties for Data Guard. This new privilege is based on the least privilege principle, in which a user is granted only the necessary privileges required to perform a specific function and no more. The SYSDBA privilege continues to work as in previous releases.
- The Oracle Database In-Memory column store is supported on standby databases in an Active Data Guard environment.
- Further improves performance and availability of Data Warehouses in a Data Guard configuration by tracking information from `NOLOGGING` operations so they can be repaired with the new RMAN command `RECOVER DATABASE NOLOGGING`.
- Improves the impact multiple SYNC transport destinations have on the primary database through the use of a new parameter `DATA_GUARD_SYNC_LATENCY`. This parameter defines the maximum amount of time (in seconds) that the Primary database must wait before disconnecting subsequent destinations after at least one synchronous standby has acknowledged receipt of the redo.
- Data Guard Broker improves manageability by supporting destinations of different Endianess than the primary in addition to enhancing management of alternate destinations.
- Data Guard improves protection and Return To Operations (RTO) and Recovery Point Objectives (RPO) through multiple features including:
  - Multi-Instance Redo Apply (MIRA) provides scalable redo apply performance across Oracle RAC instances reducing RTO for even higher production OLTP or batch workloads
  - Compare primary and standby database blocks using the new `DBMS_DBCOMP` package to help identify lost writes so they can be resolved efficiently.
  - Fast Start Failover (FSFO) has the robustness of highly available zero data loss configurations with support for Maximum Protection mode while giving the flexibility of multiple observers and multiple failover targets for high availability in any configuration. FSFO can also be configured to automatically fail over to the standby with the detection of a lost write on the primary .
  - RPO is improved with no data loss failovers after a storage failure in ASYNC configurations and Data Guard Broker support for Application Continuity, improving the user experience during Data Guard role transitions.
- Oracle Data Guard Broker further improves the management of databases by supporting destinations of different endianness than the primary in addition to enhancing management of alternate archive destinations when the primary destination is unavailable.
- Oracle Data Guard Database Compare tool compares data blocks stored in an Oracle Data Guard primary database and its physical standby databases. Use this tool to find disk errors (such as lost write) that cannot be detected by other tools like the DBVERIFY utility. (new in Oracle Database 12c Release 2)
- Oracle Data Guard Broker supports multiple automatic failover targets in a fast-start failover configuration. Designating multiple failover targets significantly improves the likelihood that there is always a standby suitable for automatic failover when needed. (new in Oracle Database 12c Release 2)

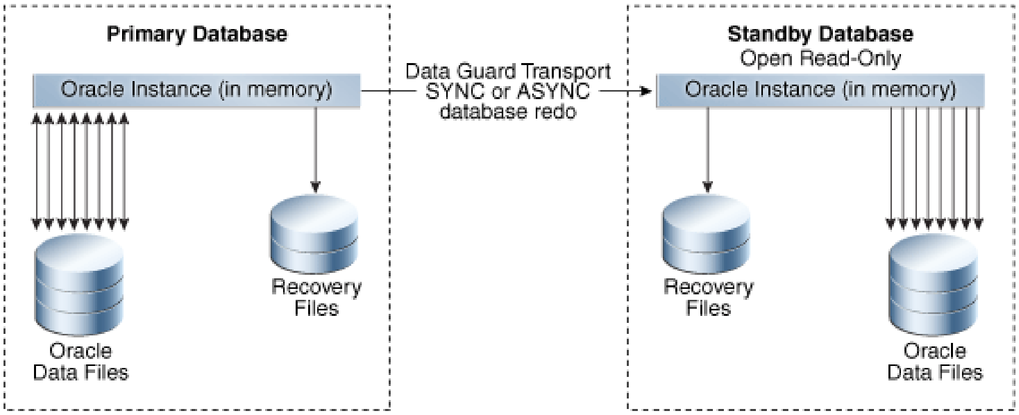
- Dynamically change Oracle Data Guard Broker Fast-Start Failover target. The fast-start failover target standby database can be changed dynamically, to another standby database in the target list, without disabling fast-start failover. (new in Oracle Database 19c)
- Propagate restore points from primary to standby Site. Restore points created on the primary database are propagated to the standby sites, so that they are available even after a failover operation. (new in Oracle Database 19c)
- Oracle Data Guard automatic outage resolution can be tuned to fit your specific needs. Oracle Data Guard has an internal mechanism to detect hung processes and terminate them, allowing the normal outage resolution to occur. (new in Oracle Database 19c)
- Active Data Guard DML redirection helps load balancing between the primary and standby databases. Incidental Data Manipulation Language (DML) operations can be run on Active Data Guard standby databases. This allows more applications to benefit from using an Active Data Guard standby database when some writes are required. When incidental DML is issued on an Active Data Guard standby database, the update is passed to the primary database where it is executed. The resulting redo of the transaction updates the standby database after which control is returned to the application. (new in Oracle Database 19c)

Oracle Active Data Guard

Oracle Active Data Guard is Oracle's strategic solution for real time data protection and disaster recovery for the Oracle database using a physical replication process.

Oracle Active Data Guard also provides high return on investment in disaster recovery systems by enabling a standby database to be open read-only while it applies changes received from the primary database. Oracle Active Data Guard is a separately licensed product that provides advanced features that greatly expand Data Guard capabilities included with Oracle Enterprise Edition.

Figure 3-1 Oracle Active Data Guard Architecture



Oracle Active Data Guard enables administrators to improve performance by offloading processing from the primary database to a physical standby database that is open read-only while it applies updates received from the primary database. Offload capabilities of Oracle Active Data Guard include read-only reporting and ad-hoc queries (including DML to global temporary tables and unique global or session sequences), data extracts, fast incremental backups, redo transport compression, efficient servicing of multiple remote destinations, and the ability to extend zero data loss protection to a remote standby database without impacting primary database performance. Oracle Active Data Guard also increases high availability by performing automatic block repair and enabling High Availability Upgrades automation.

Note:

Oracle Active Data Guard is licensed separately as a database option license for Oracle Database Enterprise Edition. All Oracle Active Data Guard capabilities are also included in an Oracle Golden Gate license for Oracle Enterprise Edition. This provides customers with the choice of a standalone license for Oracle Active Data Guard, or licensing Oracle GoldenGate to acquire access to all advanced Oracle replication capabilities.

Benefits of Oracle Active Data Guard

Oracle Active Data Guard inherits all of the benefits previously listed for Data Guard, plus the following:

- Improves primary database performance: Production-offload to an Oracle Active Data Guard standby database of read-only applications, reporting, and ad hoc queries. Any application compatible with a read-only database can run on an Oracle Active Data Guard standby. Oracle also provides integration that enables the offloading of many Oracle E-Business Suite Reports, PeopleTools reporting, Oracle Business Intelligence Enterprise Edition (OBIEE), and Oracle TopLink applications to an Oracle Active Data Guard standby database.
- DML global temporary tables and the use of sequences at the standby database significantly expands the number of read-only applications that can be off-loaded from production databases to an Oracle Active Data Guard standby database.
- The unique ability to easily scale read performance using multiple Oracle Active Data Guard standby databases, also referred to as a Reader Farm.
- Production-offload of data extracts using Oracle Data Pump or other methods that read directly from the source database.
- Production-offload of the performance impact from network latency in a synchronous, zero data loss configuration where primary and standby databases are separated by hundreds or thousands of miles. Far sync uses a lightweight instance (control file and archive log files, but no recovery and no data files), deployed on a system independent of the primary database. The far sync instance is ideally located at the maximum distance from the primary system that an application can tolerate the performance impact of synchronous transport to provide optimal protection. Data Guard transmits redo synchronously to the far sync instance and far sync forwards the redo asynchronously to a remote standby database that is the ultimate failover target. If the primary database fails, the same failover command used for any Data Guard configuration, or mouse click using Oracle Enterprise Manager Cloud Control, or automatic failover using Data Guard Fast-Start Failover executes a zero data loss failover to the remote destination. This transparently extends zero data loss protection to a remote standby database just as if it were receiving redo directly from the primary database, while avoiding the performance impact to the primary database of WAN network latency in a synchronous configuration.
- Production-offload of the overhead of servicing multiple remote standby destinations using far sync. In a far sync configuration, the primary database ships a single stream of redo to a far sync instance using synchronous or asynchronous transport. The far sync instance is able to forward redo asynchronously to as many as 29 remote destinations with zero incremental overhead on the source database.

- Data Guard maximum availability supports the use of the

  ```
  NOAFFIRM
  ```

  redo transport attribute. A standby database returns receipt acknowledgment to its primary database as soon as redo is received in memory. The standby database does not wait for the Remote File Server (RFS) to write to a standby redo log file.

  This feature provides increased primary database performance in Data Guard configurations using maximum availability and SYNC redo transport. Fast Sync isolates the primary database in a maximum availability configuration from any performance impact due to slow I/O at a standby database. This new FAST SYNC feature can work with a physical standby target or within a far sync configuration.
- Production-offload of CPU cycles required to perform redo transport compression. Redo transport compression can be performed by the far sync instance if the Data Guard configuration is licensed for Oracle Advanced Compression. This conserves bandwidth with zero incremental overhead on the primary database.
- Production-offload and increased backup performance by moving fast incremental backups off of the primary database and to the standby database by utilizing Oracle Active Data Guard support for RMAN block change tracking.
- Increased high availability using Oracle Active Data Guard automatic block repair to repair block corruptions, including file header corruptions, detected at either the primary or standby, transparent to applications and users.
- Increased high availability by reducing planned downtime for upgrading to new Oracle Database patch sets and database releases using the additional automation provided by high availability Upgrade.
- Connection preservation on an Active Data Guard standby through a role change facilitates improved reporting and improves the user experience. The connections pause while the database role changes to a primary database and resume, improving the user experience.
- The Oracle Enterprise Manager Diagnostic tool can be used with Active Data Guard to capture and send performance data to the Automatic Workload Repository, while the SQL Tuning Advisor allows primary database SQL statement tuning to be offloaded to a standby database.
- Active Data Guard support for the Oracle Database In-Memory option enables reporting to be offloaded to the standby database while reaping the benefits the In-Memory option provides, including tailored column stores for the standby database workload.

Oracle Data Guard Advantages Over Traditional Solutions

Oracle Data Guard provides a number of advantages over traditional solutions.

- Fast, automatic or automated database failover for data corruptions, lost writes, and database and site failures, with recovery times of potentially seconds with Data Guard as opposed to hours with traditional solutions
- Zero data loss over wide area network using Oracle Active Data Guard Far Sync
- Offload processing for redo transport compression and redo transmission to up to 29 remote destinations using Oracle Active Data Guard Far Sync

- Automatic corruption repair automatically replaces a physical block corruption on the primary or physical standby by copying a good block from a physical standby or primary database
- Most comprehensive protection against data corruptions and lost writes on the primary database
- Reduced downtime for storage, Oracle ASM, Oracle RAC, system migrations and some platform migrations, and changes using Data Guard switchover
- Reduced downtime for database upgrades with Data Guard rolling upgrade capabilities
- Ability to off-load primary database activities—such as backups, queries, or reporting—without sacrificing the RTO and RPO ability to use the standby database as a read-only resource using the real-time query apply lag capability, including Database In-Memory column support
- Ability to integrate non-database files using Oracle Database File System (DBFS) or Oracle Automatic Storage Management Cluster File System (Oracle ACFS) as part of the full site failover operations
- No need for instance restart, storage remastering, or application reconnection after site failures
- Transparency to applications
- Transparent and integrated support (application continuity and transaction guard) for application failover
- Effective network utilization
- Database In-Memory support
- Integrated service and client failover that reduces overall application RTO
- Enhanced and integrated Data Guard awareness with existing Oracle technologies such as Oracle RAC, RMAN, Oracle GoldenGate, Enterprise Manager, health check (`orachk`), DBCA, and Fleet Patch and Provisioning

For data resident in Oracle databases, Data Guard, with its built-in zero-data-loss capability, is more efficient, less expensive, and better optimized for data protection and disaster recovery than traditional remote mirroring solutions. Data Guard provides a compelling set of technical and business reasons that justify its adoption as the disaster recovery and data protection technology of choice, over traditional remote mirroring solutions.

Data Guard and Planned Maintenance

Data Guard standby databases can be used to reduce planned downtime by performing maintenance in a rolling fashion. Changes are implemented first at the standby database. The configuration is allowed to run with the primary at the old version and standby at the new version until there is confidence that the new version is ready for production. A Data Guard switchover can be performed, transitioning production to the new version or same changes can be applied to production in a rolling fashion. The only possible database downtime is the time required to perform the switchover.

There are several approaches to performing maintenance in a rolling fashion using a Data Guard standby. Customer requirements and preferences determine which approach is used.

Data Guard Redo Apply and Standby-First Patching

Beginning with Oracle Database 10g, there has been increased flexibility in cross-platform support using Data Guard Redo Apply.

In certain Data Guard configurations, primary and standby databases are able to run on systems having different operating systems (for example, Windows and Linux), word size (32bit/64bit), different storage, different Exadata hardware and software versions, or different hardware architectures. Redo Apply can also be used to migrate to Oracle Automatic Storage Management (ASM), to move from single instance Oracle databases to Oracle RAC, to perform technology refresh, or to move from one data center to the next.

Beginning with Oracle Database 11g Release 2 (11.2), Standby-First Patch Apply (physical standby using Redo Apply) can support different database software patch levels between a primary database and its physical standby database for the purpose of applying and validating Oracle patches in a rolling fashion. Patches eligible for Standby-First patching include:

- Database Release Updates (RUs) or Release Update Revisions (RURs)
- Database Patch Set Update (PSU)
- Database Critical Patch Update (CPU)
- Database bundled patch

Standby-First Patch Apply is supported for certified database software patches for Oracle Database Enterprise Edition 11g Release 2 (11.2) and later.

In each of the types of planned maintenance previously described, the configuration begins with a primary and physical standby database (in the case of migration to a new platform, or to ASM or Oracle RAC, the standby is created on the new platform). After all changes are implemented at the physical standby database, Redo Apply (physical replication) is used to synchronize the standby with the primary. A Data Guard switchover is used to transfer production to the standby (the new environment).

See Also:

My Oracle Support Note 413484.1 for information about mixed platform combinations supported in a Data Guard configuration.

My Oracle Support Note 1265700.1 for more information about Standby First Patch Apply and the README for each patch to determine if a target patch is certified as being a Standby-First Patch.

Data Guard Transient Logical Rolling Upgrades

There are numerous types of maintenance tasks that are unable to use Redo Apply (physical replication) to synchronize the original version of a database with the changed or upgraded version. These tasks include:

- Database patches or upgrades that are not Standby-First Patch Apply-eligible. This includes database patch-sets (11.2.0.2 to 11.2.0.4) and upgrade to new Oracle Database releases (18c to 19c).

- Maintenance must be performed that modifies the physical structure of a database that would require downtime (for example, adding partitioning to non-partitioned tables, changing Basicfile LOBs to Securefile LOBs, changing XML-CLOB to Binary XML, or altering a table to be OLTP-compressed).

All of the previous types of maintenance can be performed in a rolling fashion using a Data Guard standby database by using Data Guard SQL Apply (logical replication) to synchronize the old and new versions of the database. Prior to Oracle Database 11g this required creating a logical standby database, performing the maintenance on the logical standby, resynchronizing the standby with the primary, and then switching over. Additionally if a physical standby was being used for disaster recovery, then a new physical standby database would have to be created from a backup of the production database at the new version. This represented a number of logistical and cost challenges when upgrading a multi-terabyte database.

Beginning with Oracle Database 11g, database rolling upgrades can use a new procedure called Transient Logical that begins and ends with a physical standby database. SQL Apply is only used during the phase when Data Guard is synchronizing across old and new versions. A new logical standby database does not need to be created if there is already a physical standby in place. A new physical standby database does not need to be created from a backup of the production database at the new version after the maintenance is complete. Similar to the traditional process of upgrading a Data Guard configuration having an in-place physical standby, the original primary is upgraded or changed using redo from the new primary database and Redo Apply (a single catalog upgrade migrates both primary and standby databases to the new Oracle release).

Transient Logical upgrades require that the primary database be at Oracle Database 11g release 1 (11.1) or later and that the database meet the prerequisites of SQL Apply.

Oracle provides a Bourne shell script that automates a number of the manual steps required by the Transient Logical rolling upgrade process.

Databases that use Oracle Database Vault can be upgraded to new Oracle Database releases and patch sets by using Oracle Data Guard database rolling upgrades (transient logical standby only).

See Also:

http://www.oracle.com/goto/maa for Oracle MAA white paper “Oracle Database Rolling Upgrades: Using a Data Guard Physical Standby Database”

Rolling Upgrade Using Oracle Active Data Guard

Rolling database upgrade using Oracle Active Data Guard provides a simpler, automated, and easily repeatable method for reducing planned downtime than represented by the manual Transient Logical rolling upgrade procedure.

Rolling upgrade using Oracle Active Data Guard transforms the 42 or more steps required by the manual procedure into several easy-to-use DBMS_ROLLING PL/SQL packages. Rolling upgrades performed using the DBMS_ROLLING PL/SQL package are supported on a multitenant container database (CDB).

A rolling upgrade using Oracle Active Data Guard:

- Generates an upgrade plan with a configuration-specific set of instructions to guide you through the upgrade process.
- Modifies parameters of the rolling upgrade.
- Configures primary and standby databases participating in the upgrade.
- Performs switchover of the production database to the new version. Switchover is the only downtime required.
- Completes the upgrade of the old primary and any additional standby databases in the Data Guard configuration and resynchronizes with the new primary.

Rolling upgrade using Oracle Active Data Guard has the following additional benefits:

- Provides a simple specify-compile-execute protocol
  - Catches configuration errors at the compilation step
  - Runtime errors are detected during execution
- The state is kept in the database
  - Enables a reliable, repeatable process
- Runtime steps are constant regardless of how many databases are involved
- Handles failure at the original primary database
- Enables data protection for the upgraded primary at all times

> **See Also:**
>
> http://www.oracle.com/goto/maa for Oracle MAA white paper “Oracle Database Rolling Upgrades: Using a Data Guard Physical Standby Database”
>
> *Oracle Data Guard Concepts and Administration*

Oracle GoldenGate

Oracle GoldenGate is Oracle's strategic logical replication solution for data distribution and data integration.

Oracle GoldenGate offers a real-time, log-based change data capture and replication software platform. The software provides capture, routing, transformation, and delivery of transactional data across heterogeneous databases in real time.

Unlike replication solutions from other vendors, Oracle GoldenGate is more closely integrated with Oracle Database while also providing an open, modular architecture ideal for replication across heterogeneous database management systems. This combination of attributes eliminates compromise, making Oracle GoldenGate the preferred logical replication solution for addressing requirements that span Oracle Database and non-Oracle Database environments.

A typical environment includes a capture, pump, and delivery process. Each of these processes can run on most of the popular operating systems and databases, including Oracle Database. All or a portion of the data can be replicated, and the data within any of these processes can be manipulated for not only heterogeneous environments but also different database schemas, table names, or table structures. Oracle GoldenGate also supports bidirectional replication with preconfigured conflict detection and resolution handlers to aid in resolving data conflicts.

Oracle GoldenGate logical replication enables all databases in an Oracle GoldenGate configuration, both source and target databases, to be open read-write. This makes it a key component of MAA for addressing a broad range of high availability challenges for zero downtime maintenance, cross platform migration, and continuous data availability, specifically:

- **Zero or near zero downtime maintenance.** In this architecture, Oracle GoldenGate provides greater flexibility than the capabilities provided by Data Guard. Oracle GoldenGate source and target databases can have a different physical and logical structure, can reside on different hardware and operating system architectures, can span wide differences in Oracle Database releases (for example, 12.2 to 19c), or be a mix of Oracle and non-Oracle systems. This allows for the modernization of 24x7 servers and allows new Oracle features to be implemented without impacting the availability of the databases. Maintenance is first performed on a target database while production runs on the source. After the maintenance is complete, production can be moved to the source all at once, similar to a Data Guard switchover. Optionally, bidirectional replication can be used to gradually move users over to the new system to create the perception of zero downtime. In either case, Oracle GoldenGate replication can be enabled in the reverse direction to keep the original source database synchronized during a transition period, making it simple to effect a planned fall-back to the previous version if needed, with minimal downtime and no data loss.
- **Zero or near-zero downtime migrations when a Data Guard solution is not applicable.** Platform or database migrations can be carried out using Oracle GoldenGate as the data synchronization method between the old and new systems. Once the database has been instantiated on another host, Oracle GoldenGate is configured to replicate changes from the production database. A guaranteed restore point can be created on the migrated database so that after user testing the database can be flashed back, and Oracle GoldenGate can apply any outstanding data changes from the production database before moving the application users to the new database, similar to a snapshot standby database. If desired, bi-directional replication can also be configured from the migrated database back to the production database for use as a fallback solution.
- **Zero or near-zero downtime application upgrades.** Application upgrades that modify back-end database objects typically result in significant planned downtime while maintenance is being performed. Oracle GoldenGate replication enables data transformations that map database objects used by a previous version of an application to objects modified by the new version of an application. This enables database maintenance to be performed on a separate copy of the production database without impacting the availability of the application. After the maintenance is complete and Oracle GoldenGate has finished synchronizing old and new versions, users can be switched to the new version of the application.
- **Read-write access to a replica database while it is being synchronized with its source database.** This is most often used to offload reporting to a copy of a production database when the reporting application requires a read-write connection to database in order to function. This is also relevant to disaster recovery environments where the nature of the technology used for the application tier requires an active read-write connection to the DR database at all times in order to meet recovery time objectives.
- **Active-Active replication.** Oracle GoldenGate supports an active-active multi-directional configuration, where there are two or more systems with identical sets of data that can be changed by application users on either system. Oracle GoldenGate replicates

transactional data changes from each database to the others to keep all sets of data current.

- Seamless moves between Oracle Real Application Clusters (RAC) nodes in the event of database instance failure or during applicable maintenance operations. This ability provides high availability with Oracle GoldenGate and it is possible to patch and upgrade the Oracle GoldenGate software on one or more nodes in the cluster without affecting the node where Oracle GoldenGate is currently running. Then at a predetermined time, Oracle GoldenGate can be switched to one of the upgraded nodes. The switch is done without reconfiguring Oracle GoldenGate because configuration information is shared across the Oracle RAC cluster.

> **See Also:**
>
> Oracle GoldenGate Documentation
>
> http://www.oracle.com/goto/maa for Oracle MAA Oracle GoldenGate white papers

Best Practice: Oracle Active Data Guard and Oracle GoldenGate

While Oracle Active Data Guard and Oracle GoldenGate are each capable of maintaining a synchronized copy of an Oracle database, each has unique characteristics that result in high availability architectures that can use one technology or the other, or both at the same time, depending upon requirements.

Examples of MAA Best Practice guidelines are as follows:

When to Use Oracle Active Data Guard

Use Oracle Active Data Guard when the emphasis is on simplicity, data protection, and availability.

- Simplest, fastest, one-way replication of a complete Oracle database.
- No restrictions: Data Guard Redo Apply supports all data and storage types and Oracle features; transparent replication of DDL
- Features optimized for data protection: Detects silent corruptions that can occur on source or target; automatically repairs corrupt blocks
- Synchronized standby open read-only provides simple read-only offloading for maximum ROI
- Transparency of backups: A Data Guard primary and standby are physically exact copies of each other; RMAN backups are completely interchangeable
- Zero data loss protection at any distance, without impacting database performance
- Minimizing planned downtime and risk using standby first patching, database rolling upgrades, and select platform migrations
- Reduce risk of introducing change by dual purposing a DR system for testing using Data Guard Snapshot Standby

- Integrated automatic database and client failover
- Integrated management of a complete configuration: Data Guard Broker command line interface or Oracle Enterprise Manager Cloud Control

When to Use Oracle GoldenGate

Use Oracle GoldenGate when the emphasis is on advanced replication requirements not addressed by Oracle Active Data Guard.

- Any requirement where the replica database must be open read/write while synchronizing with the primary database
- Any data replication requirements such as multimaster and bidirectional replication, subset replication, many-to-one replication, and data transformations.
- When data replication is required between endian format platforms or across-database major versions.
- Maintenance and migrations where zero downtime or near zero downtime is required. Oracle GoldenGate can be used to migrate between application versions, for example, from Application 1.0 to Application 2.0 without downtime.
- Database rolling upgrades where it is desired to replicate from new version down to the old version for the purpose of fast fall-back if something is wrong with the upgrade.
- Zero downtime planned maintenance where bidirectional replication is used to gradually migrate users to the new version, creating the perception of zero downtime. Note that bidirectional replication requires avoiding or resolving update conflicts that can occur on disparate databases.

When to Use Oracle Active Data Guard and Oracle GoldenGate Together

Oracle Active Data Guard and Oracle GoldenGate are not mutually exclusive. The following are use cases of high availability architectures that include the simultaneous use of Oracle Active Data Guard and Oracle GoldenGate.

- An Oracle Active Data Guard standby is utilized for disaster protection and database rolling upgrades for a mission critical OLTP database. At the same time, Oracle GoldenGate is used to replicate data from the Data Guard primary database (or from the standby database using Oracle GoldenGate ALO mode) for ETL update of an enterprise data warehouse.
- Oracle GoldenGate subset replication is used to create an operational data store (ODS) that extracts, transforms, and aggregates data from numerous data sources. The ODS supports mission critical application systems that generate significant revenue for the company. An Oracle Active Data Guard standby database is used to protect the ODS, providing optimal data protection and availability.
- Oracle GoldenGate bidirectional replication is utilized to synchronize two databases separated by thousands of miles. User workload is distributed across each database based upon geography, workload, and service level using Global Data Services (GDS). Each Oracle GoldenGate copy has its own local synchronous Data Guard standby database that enables zero data loss failover if an outage occurs. Oracle GoldenGate capture and apply processes are easily restarted on the new primary database following a failover because the primary and standby are an exact, up-to-date replica of each other.
- An Oracle Active Data Guard standby database used for disaster protection is temporarily converted into an Oracle GoldenGate target for the purpose of performing

planned maintenance not supported by Data Guard. For example, a Siebel application upgrade requiring modification of back-end database objects which require comprehensive testing before switching users over to the new system.

- Oracle Active Data Guard is used to protect a production environment when a major database version upgrade is required offering zero or near-zero downtime (for example, Oracle 18c to 19c.) A second primary/standby environment is created using the new database version, and Oracle GoldenGate is used to replicate data from the production environment to the copy with one-way or bidirectional replication. When Oracle GoldenGate has completed synchronizing the old and new environments, production is switched to the new environment and the old environment is decommissioned. This provides zero or minimal downtime depending upon configuration, eliminates risk by providing complete isolation between the old and new environment, and avoids any impact to data protection and availability SLAs if problems are encountered during the upgrade process.

> **See Also:**
>
> http://www.oracle.com/goto/maa for Oracle MAA Best Practices white paper ““Transparent Role Transitions With Oracle Data Guard and Oracle GoldenGate"

Recovery Manager

Recovery Manager (RMAN) provides a comprehensive foundation for efficiently backing up and recovering the database. RMAN eliminates operational complexity while providing superior performance and availability of the database.

RMAN determines the most efficient method of executing the requested backup, restoration, or recovery operation and then submits these operations to the Oracle Database server for processing. RMAN and the server automatically identify modifications to the structure of the database and dynamically adjust the required operation to adapt to the changes.

RMAN is the standard interface to backup and restore from Recovery Appliance, local disk (ZFS storage), tape, and cloud object store.

RMAN provides the following benefits:

- Support for Oracle Sharding - RMAN support for every independent database (shard)
- Enhancement for Sparse Databases - allows backup and restore to operate on

  ```
  SPARSE
  ```

  backup sets and or image copies
- Over the Network Standby Database repair of

  ```
  NONLOGGED
  ```

operation - new syntax for validation and repair on Standby -

```
VALIDATE/RECOVER .. NONLOGGED BLOCK;
```

-
  ```
  RMAN DUPLICATE
  ```

  feature enhanced to support creation of Far Sync from Primary and backup

-
  ```
  RMAN DUPLICATE
  ```

  Using Encrypted Backups - RMAN enhanced support non Auto-login wallet based encrypted backups with a new

  ```
  SET
  ```

  command - enables interrupt-free cloning
- Support for cross-platform backup and restore over the network
- Network-enabled restoration allows the

  ```
  RESTORE
  ```

  operations to copy data files directly from one database to another over the network
- Simplified table restoration with the

  ```
  RECOVER TABLE
  ```

  command
- Support for Oracle Multitenant, including backup and recovery of individual pluggable databases
- Support for cross-platform Oracle Multitenant, including backup and recovery of individual PDBs
- Automatic channel failover on backup and restore operations
- Automatic failover to a previous backup when the restore operation discovers a missing or corrupt backup
- Automatic creation of new database files and temporary files during recovery
- Automatic recovery through a previous point-in-time recovery—recovery through reset logs
- Block media recovery, which enables the data file to remain online while fixing the block corruption
- Fast incremental backups using block change tracking
- Fast backup and restore operations with intrafile and interfile parallelism
- Enhanced security with a virtual private recovery catalog
- Merger of incremental backups into image copies, providing up-to-date recoverability
- Optimized backup and restoration of required files only

- Retention policy to ensure that relevant backups are retained
- Ability to resume backup and restore operations in case of failure
- Automatic backup of the control file and the server parameter file, ensuring that backup metadata is available in times of database structural changes and media failure and disasters
- Easily reinstantiate a new database from an existing backup or directly from the production database (thus eliminating staging areas) using the

  ```
  DUPLICATE
  ```

  command.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide*

Oracle Real Application Clusters and Oracle Clusterware

Oracle RAC and Oracle Clusterware enable Oracle Database to run any packaged or custom application across a set of clustered servers.

This capability provides the highest levels of availability and the most flexible scalability. If a clustered server fails, then Oracle Database continues running on the surviving servers. When more processing power is needed, you can add another server without interrupting access to data.

Oracle RAC enables multiple instances that are linked by an interconnect to share access to an Oracle database. In an Oracle RAC environment, Oracle Database runs on two or more systems in a cluster while concurrently accessing a single shared database. The result is a single database system that spans multiple hardware systems, enabling Oracle RAC to provide high availability and redundancy during failures in the cluster. Oracle RAC accommodates all system types, from read-only data warehouse systems to update-intensive online transaction processing (OLTP) systems.

Oracle Clusterware is software that, when installed on servers running the same operating system, enables the servers to be bound together to operate as if they are one server, and manages the availability of user applications and Oracle databases. Oracle Clusterware also provides all of the features required for cluster management, including node membership, group services, global resource management, and high availability functions:

- For high availability, you can place Oracle databases (single-instance or Oracle RAC databases), and user applications (Oracle and non-Oracle) under the management and protection of Oracle Clusterware so that the databases and applications restart when a process fails or so that a failover to another node occurs after a node failure.
- For cluster management, Oracle Clusterware presents multiple independent servers as if they are a single-system image or one virtual server. This single virtual server is preserved across the cluster for all management operations,

enabling administrators to perform installations, configurations, backups, upgrades, and monitoring functions. Then, Oracle Clusterware automatically distributes the execution of these management functions to the appropriate nodes in the cluster.

Oracle Clusterware is a requirement for using Oracle RAC. Oracle Clusterware is the only clusterware that you need for most platforms on which Oracle RAC operates. Although Oracle Database continues to support third-party clusterware products on specified platforms, using Oracle Clusterware provides these main benefits:

- Dispenses with proprietary vendor clusterware
- Uses an integrated software stack from Oracle that provides disk management with local or remote Oracle Automatic Storage Management (Oracle Flex ASM) to data management with Oracle Database and Oracle RAC
- Can be configured in large clusters, called an Oracle Flex Cluster.

In addition, Oracle Database features, such as Oracle services, use the underlying Oracle Clusterware mechanisms to provide their capabilities.

Oracle Clusterware requires two clusterware components: a voting disk to record node membership information and the Oracle Cluster Registry (OCR) to record cluster configuration information. The voting disk and the OCR must reside on shared storage. Oracle Clusterware requires that each node be connected to a private network over a private interconnect.

Benefits of Using Oracle Clusterware

Oracle Clusterware provides the following benefits.

- Tolerates and quickly recovers from computer and instance failures.
- Simplifies management and support by means of using Oracle Clusterware together with Oracle Database. By using fewer vendors and an all Oracle stack you gain better integration compared to using third-party clusterware.
- Performs rolling upgrades for system and hardware changes. For example, you can apply Oracle Clusterware upgrades, patch sets, and interim patches in a rolling fashion.

  When you upgrade to Oracle Database 12c, Oracle Clusterware and Oracle ASM binaries are installed as a single binary called the Oracle Grid Infrastructure. You can upgrade Oracle Clusterware in a rolling manner from Oracle Clusterware 10g and Oracle Clusterware 11g; however, you can only upgrade Oracle ASM in a rolling manner from Oracle Database 11g release 1 (11.1).
- Automatically restarts failed Oracle processes.
- Automatically manages the virtual IP (VIP) address. When a node fails, the node's VIP address fails over to another node on which the VIP address can accept connections.
- Automatically restarts resources from failed nodes on surviving nodes.
- Controls Oracle processes as follows:
  - For Oracle RAC databases, Oracle Clusterware controls all Oracle processes by default.
  - For Oracle single-instance databases, Oracle Clusterware enables you to configure the Oracle processes into a resource group that is under the control of Oracle Clusterware.

- Provides an application programming interface (API) for Oracle and non-Oracle applications that enables you to control other Oracle processes with Oracle Clusterware, such as restart or react to failures and certain rules.
- Manages node membership and prevents split-brain syndrome in which two or more instances attempt to control the database.
- Using server weight-based node eviction allows for aligning the choice of which node gets evicted in case of certain failures in the cluster with business requirements, ensuring that the most important workload is kept alive for as long as possible, assuming an equal choice between servers.
- Provides the ability to perform rolling release upgrades of Oracle Clusterware, with no downtime for applications.

Benefits of Using Oracle Real Application Clusters and Oracle Clusterware

Together, Oracle RAC and Oracle Clusterware provide all of the Oracle Clusterware benefits plus the following benefits.

- Provides better integration and support of Oracle Database by using an all Oracle software stack compared to using third-party clusterware.
- Relocate Oracle Service automatically. Plus, when you perform additional fast application notification (FAN) and client configuration, distribute FAN events so that applications can react immediately to achieve fast, automatic, and intelligent connection and failover.
- Detect connection failures fast and automatically, and remove terminated connections for any Java application using Oracle Universal Connection Pool (Oracle UCP) Fast Connection Failover and FAN events.
- Balance work requests using Oracle UCP runtime connection load balancing.
- Use runtime connection load balancing with Oracle UCP, Oracle Call Interface (OCI), and Oracle Data Provider for .NET (ODP.NET).
- Distribute work across all available instances using load balancing advisory.
- You can configure a database so that Oracle Clusterware is aware of the CPU requirements and limits for the given database. Oracle Clusterware uses this information to place the database resource only on servers that have a sufficient number of CPUs, amount of memory, or both.
- Allow the flexibility to increase processing capacity using commodity hardware without downtime or changes to the application.
- Provide comprehensive manageability integrating database and cluster features.
- Provide scalability across database instances.
- Implement Fast Connection Failover for nonpooled connections.

Oracle RAC Advantages Over Traditional Cold Cluster Solutions

Oracle RAC provides many advantages over traditional cold cluster solutions, including the following.

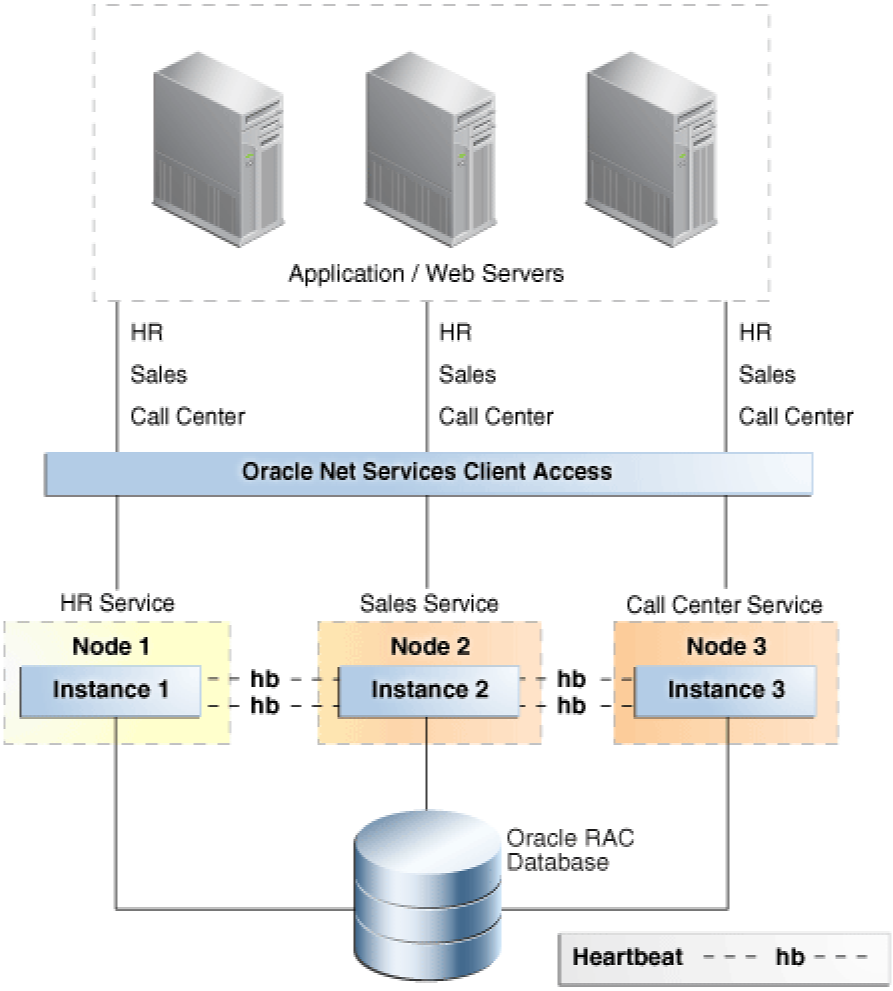
- Scalability across database instances

- Flexibility to increase processing capacity using commodity hardware without downtime or changes to the application
- Ability to tolerate and quickly recover from computer and instance failures (measured in seconds)
- Application brownout can be zero or seconds compared to minutes and hours with cold cluster solutions
- Optimized communication in the cluster over redundant network interfaces, without using bonding or other technologies

  Oracle Grid Infrastructure and Oracle RAC make use of Redundant Interconnect Usage that distributes network traffic and ensures optimal communication in the cluster. This functionality is available starting with Oracle Database 11g Release 2 (11.2.0.2). In previous releases, technologies like bonding or trunking were used to make use of redundant networks for the interconnect.
- Rolling upgrades for system and hardware changes
- Rolling patch upgrades for some interim patches, security patches, CPUs, and cluster software
- Fast, automatic, and intelligent connection and service relocation and failover
- Comprehensive manageability integrating database and cluster features with Grid Plug and Play and policy-based cluster and capacity management
- Load balancing advisory and run-time connection load balancing help redirect and balance work across the appropriate resources
- Oracle Quality of Service (QoS) Management for policy-based run-time management of resource allocation to database workloads to ensure service levels are met in order of business need under dynamic conditions. This is accomplished by assigning a service to a server pool where the database is running. Resources from the pool are used to make sure the required capacity is available.
- Oracle Enterprise Management support for Oracle ASM and Oracle ACFS, Grid Plug and Play, Cluster Resource Management, Oracle Clusterware and Oracle RAC Provisioning and patching.
- SCAN (Single Client Access Name) support as a single name to the clients connecting to Oracle RAC that does not change throughout the life of the cluster, even if you add or remove nodes from the cluster.

The following figure shows Oracle Database with Oracle RAC architecture. This figure shows Oracle Database with Oracle RAC architecture for a partitioned three-node database. An Oracle RAC database is connected to three instances on different nodes. Each instance is associated with a service: HR, Sales, and Call Center. The instances monitor each other by checking "heartbeats." Oracle Net Services provide client access to the Application/web server tier at the top of the figure.

Figure 3-2   Oracle Database with Oracle RAC Architecture



Note:

After Oracle release 11.2, Oracle RAC One Node or Oracle RAC is the preferred solution over Oracle Clusterware (Cold Cluster Failover) because it is a more complete and feature-rich solution.

See Also:

Oracle RAC Administration and Deployment Guide

Oracle Clusterware Administration and Deployment Guide

Oracle RAC One Node

Oracle Real Application Clusters One Node (Oracle RAC One Node) is a single instance of an Oracle RAC database that runs on one node in a cluster.

This feature enables you to consolidate many databases into one cluster with minimal overhead, protecting them from both planned and unplanned downtime. The consolidated databases reap the high availability benefits of failover protection, online rolling patch application, and rolling upgrades for the operating system and Oracle Clusterware.

Oracle RAC One Node enables better availability than cold failover for single-instance databases because of the Oracle technology called online database relocation, which intelligently migrates database instances and connections to other cluster nodes for high availability and load balancing. Online database relocation is performed using the Server Control Utility (SRVCTL).

Oracle RAC One Node provides the following:

- Always available single-instance database services
- Built-in cluster failover for high availability
- Live migration of instances across servers
- Online rolling patches and rolling upgrades for single-instance databases
- Online upgrade from single-instance to multiple-instance Oracle RAC
- Better consolidation for database servers
- Enhanced server virtualization
- Lower cost development and test platform for full Oracle RAC
- Relocation of Oracle RAC primary and standby databases configured with Data Guard. This functionality is available starting with Oracle Database 11g Release 2 (11.2.0.2).

Oracle RAC One Node also facilitates the consolidation of database storage, standardizes your database environment, and, when necessary, enables you to transition to a full, multiple-instance Oracle RAC database without downtime or disruption.

Oracle Automatic Storage Management

Oracle ASM provides a vertically integrated file system and volume manager directly in the Oracle Database kernel.

This design provides several benefits, resulting in:

- Significantly less work to provision database storage
- Higher level of availability
- Elimination of the expense, installation, and maintenance of specialized storage products
- Unique capabilities for database applications

For optimal performance, Oracle ASM spreads files across all available storage. To protect against data loss, Oracle ASM extends the concept of SAME (stripe and mirror everything) and adds more flexibility because it can mirror at the database file level rather than at the entire disk level.

More important, Oracle ASM simplifies the processes of setting up mirroring, adding disks, and removing disks. Instead of managing hundreds or possibly thousands of files (as in a large data warehouse), database administrators using Oracle ASM create and administer a larger-grained object called a disk group. The disk group identifies the set of disks that are managed as a logical unit. Automation of file naming and placement of the underlying database files save administrators time and ensure adherence to standard best practices.

The Oracle ASM native mirroring mechanism (two-way or three-way) protects against storage failures. With Oracle ASM mirroring, you can provide an additional level of data protection with the use of failure groups. A failure group is a set of disks sharing a common resource (disk controller or an entire disk array) whose failure can be tolerated. After it is defined, an Oracle ASM failure group intelligently places redundant copies of the data in separate failure groups. This ensures that the data is available and transparently protected against the failure of any component in the storage subsystem.

By using Oracle ASM, you can:

- Mirror and stripe across drives and storage arrays.
- Automatically remirror from a failed drive to remaining drives.
- Automatically rebalance stored data when disks are added or removed while the database remains online.
- Support Oracle database files and non-database files using Oracle Automatic Storage Management Cluster File System (Oracle ACFS).
- Allow for operational simplicity in managing database storage.
- Manage the Oracle Cluster Registry (OCR) and voting disks.
- Provide preferred read capability on disks that are local to the instance, which gives better performance for an extended cluster.
- Support very large databases.
- Support Oracle ASM rolling upgrades.
- Improve availability and reliability using the Oracle ASM disk scrubbing process to find and repair logical data corruptions using mirror disks.
- Support finer granularity in tuning and security.
- Provide fast repair after a temporary disk failure through Oracle ASM Fast Mirror Resync and automatic repair of block corruptions if a good copy exists in one of the mirrors.
- Provide disaster recovery capability for the file system by enabling replication of Oracle ACFS across the network to a remote site.
- Patch the Oracle ASM instance without impacting the clients that are being serviced using Oracle Flex ASM. A database instance can be directed to access Oracle ASM metadata from another location while the current Oracle ASM instance it is connected to is taken offline for planned maintenance.
- Monitor and manage the speed and status of Oracle ASM Disk Resync and Rebalance operations.
- Bring online multiple disks simultaneously and manage performance better by controlling resync parallelism using the Oracle ASM Resync Power Limit. Recover faster after a cell or disk failure, and the instance doing the resync is failing; this is made possible by using a Disk Resync Checkpoint which enables a resync to

resume from where it was interrupted or stopped instead of starting from the beginning.

- Automatically connect database instances to another Oracle ASM instance using Oracle Flex ASM. The local database instance can still access the required metadata and data if an Oracle ASM instance fails due to an unplanned outage.
- Use flex diskgroups to prioritize high availability benefits across multiple databases all using the same diskgroup. Some of the key HA benefits are file extent redundancy, rebalance power limit, and rebalance priority. With flex diskgroups, you can set different values for the above features for different databases, resulting in prioritization across multiple databases within one diskgroup.
- Use flex diskgroups to implement quoto_groups across multiple databases sharing one diskgroup which helps in space management and protection.
- Use flex diskgroups to create point-in-time database clones using the ASM split mirror feature.
- Use preferred reads with stretch clusters to improve performance by affinitizing reads to a site.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide*

Fast Recovery Area

The fast recovery area is a unified storage location for all recovery-related files and activities in Oracle Database.

After this feature is enabled, all RMAN backups, archived redo log files, control file autobackups, flashback logs, and data file copies are automatically written to a specified file system or Oracle ASM disk group, and the management of this disk space is handled by RMAN and the database server.

Performing a backup to disk is faster because using the fast recovery area eliminates the bottleneck of writing to tape. More important, if database media recovery is required, then data file backups are readily available. Restoration and recovery time is reduced because you do not need to find a tape and a free tape device to restore the needed data files and archived redo log files.

The fast recovery area provides the following benefits:

- Unified storage location of related recovery files
- Management of the disk space allocated for recovery files, which simplifies database administration tasks
- Fast, reliable, disk-based backup and restoration

> **See Also:**
>
> *Oracle Database Backup and Recovery User’s Guide*

Corruption Prevention, Detection, and Repair

Data block corruptions can be very disruptive and challenging to repair. Corruptions can cause serious application and database downtime and data loss when encountered and worse yet it can go undetected for hours, days and even weeks leading to even longer application downtime once detected.Unfortunately, there is not one way to comprehensively prevent, detect, and repair data corruptions within the database because the source and cause of corruptions can be anywhere in memory, hardware, firmware, storage, operating system, software, or user error. Worse yet, third-party solutions that do not understand Oracle data block semantics and how Oracle changes data blocks do not prevent and detect data block corruptions well. Third party remote mirroring technologies can propagate data corruptions to the database replica (standby) leading to a double failure, data loss, and much longer downtime. Third party backup and restore solutions cannot detect corrupted backups or bad sectors until a restore or validate operation is issued, resulting in longer restore times and once again potential data loss.

Oracle MAA has a comprehensive plan to prevent, detect, and repair all forms of data block corruptions including physical block corruptions, logical block corruptions, stray writes, and lost writes. These additional safeguards provide the most comprehensive Oracle data block corruption prevention, detection, and repair solution. Details of this plan are described in the My Oracle Support note "Best Practices for Corruption Detection, Prevention, and Automatic Repair - in a Data Guard Configuration (Doc ID 1302539.1)."

The following outlines block corruption checks for various manual operational checks and runtime and background corruption checks. Database administrators and the operations team can incorporate manual checks such as running Oracle Recovery Manager (RMAN) backups, RMAN "check logical" validations, or running the `ANALYZE VALIDATE STRUCTURE` command on important objects. Manual checks are especially important to validate data that are rarely updated or queried.

Runtime checks are far superior in that they catch corruptions almost immediately or during runtime for actively queried and updated data. Runtime checks can prevent corruptions or automatically fix corruptions resulting in better data protection and higher application availability. A new background check has been introduced in Exadata to automatically scan and scrub disks intelligently with no application overhead and to automatically fix physically corrupted blocks.

Table 3-1 Summary of Block Corruption Checks

Checks	Capabilities	Physical Block Corruption	Logical Block Corruption
Manual checks	Dbverify, Analyze	Physical block checks	Logical intra-block and inter-object consistency checks
Manual checks	RMAN	Physical block checks during backup and restore operations	Intra-block logical checks
Manual checks	ASM Scrub	Physical block checks	Some logical intra-block checks

Table 3-1 (Cont.) Summary of Block Corruption Checks

Checks	Capabilities	Physical Block Corruption	Logical Block Corruption
Runtime checks	Oracle Active Data Guard	1. Continuous physical block checking at standby during transport and apply 2. Strong database isolation eliminates single point database failure 3. Automatic repair of block corruptions, including file block headers in Oracle Database 12*c* Release 2 4. Automatic database failover	1. With `DB_LOST_WRITE_PROTECT` enabled, detection of lost writes (11.2 and higher). With 11.2.0.4 and Data Guard broker, ability to shutdown the primary when lost writes are detected on the primary database. 2. With `DB_BLOCK_CHECKING` enabled on the standby, additional intra-block logical checks
Runtime checks	Database	With `DB_BLOCK_CHECKSUM`, in-memory data block and redo checksum validation	With `DB_BLOCK_CHECKING`, in-memory intra-block check validation Starting in Oracle Database 18c, and with Shadow Lost Write Protection enabled, Oracle tracks system change numbers (SCNs) for tracked data files and enables early lost write detection. When lost writes are detected, an error is returned immediately. See Shadow Lost Write Protection description following this table.
Runtime checks	ASM and ASM software mirroring (inherent in Exadata, Supercluster, and Zero Data Loss Recovery Appliance)	Implicit data corruption detection for reads and writes and automatic repair if good ASM extent block pair is available during writes	.
Runtime checks	DIX + T10 DIF	Checksum validation from operating system to HBA controller to disk (firmware). Validation for reads and writes for certified Linux, HBA and disks.	.

Table 3-1 (Cont.) Summary of Block Corruption Checks

Checks	Capabilities	Physical Block Corruption	Logical Block Corruption
Runtime checks	Hardware and Storage	Limited checks due to lack of Oracle integration. Checksum is most common.	Limited checks due to lack of Oracle integration. Checksum is most common
Runtime checks	Exadata	Comprehensive HARD checks on writes	HARD checks on writes
Background checks	Exadata	Automatic HARD disk scrub and repair. Detects and fixes bad sectors.	.

Shadow Lost Write Protection

New in Oracle Database 18c, shadow lost write protection detects a lost write before it can result in a major data corruption. You can enable shadow lost write protection for a database, a tablespace, or a data file without requiring an Oracle Data Guard standby database. Shadow lost write protection provides fast detection and immediate response to a lost write, thus minimizing the data loss that can occur in a database due to data corruption.

See Also:

Oracle Database Reference for more information about the views and initialization parameters

My Oracle Support Note 1302539.1

Data Recovery Advisor

Data Recovery Advisor automatically diagnoses persistent (on-disk) data failures, presents appropriate repair options, and runs repair operations at your request.

You can use Data Recovery Advisor to troubleshoot primary databases, logical standby databases, physical standby databases, and snapshot standby databases.

Data Recovery Advisor includes the following functionality:

- Failure diagnosis

  The first symptoms of database failure are usually error messages, alarms, trace files and dumps, and failed health checks. Assessing these symptoms can be complicated, error-prone, and time-consuming. Data Recovery Advisor automatically diagnoses data failures and informs you about them.

- Failure impact assessment

After a failure is diagnosed, you must understand its extent and assess its impact on applications before devising a repair strategy. Data Recovery Advisor automatically assesses the impact of a failure and displays it in an easily understood format.

- Repair generation

  Even if a failure was diagnosed correctly, selecting the correct repair strategy can be error-prone and stressful. Moreover, there is often a high penalty for making poor decisions in terms of increased downtime and loss of data. Data Recovery Advisor automatically determines the best repair for a set of failures and presents it to you.

- Repair feasibility checks

  Before presenting repair options, Data Recovery Advisor validates them with respect to the specific environment and availability of media components required to complete the proposed repair, including restoring files directly from the primary or standby database to complete the proposed repair.

- Repair automation

  If you accept the suggested repair option, Data Recovery Advisor automatically performs the repair, verifies that the repair was successful, and closes the appropriate failures.

- Validation of data consistency and database recoverability

  Data Recovery Advisor can validate the consistency of your data, and backups and redo stream, whenever you choose.

- Early detection of corruption

  Through Health Monitor, you can schedule periodic runs of Data Recovery Advisor diagnostic checks to detect data failures before a database process executing a transaction discovers the corruption and signals an error. Early warnings can limit the damage caused by corruption.

- Integration of data validation and repair

  Data Recovery Advisor is a single tool for data validation and repair.

> **Note:**
>
> Data Recovery Advisor only supports single-instance databases. Oracle RAC databases are not supported.

> **See Also:**
>
> *Oracle Database Backup and Recovery User’s Guide* for information about Data Recovery Advisor supported database configurations.

Oracle Flashback Technology

Oracle Flashback technology is a group of Oracle Database features that let you view past states of database, database objects, transactions or rows or to rewind the database, database objects, transactions or rows to a previous state without using point-in-time media recovery.

With flashback features, you can:

- Perform queries to show data as it looked at a previous point in time
- Perform queries that return metadata that shows a detailed history of changes to the database
- Recover tables or rows to a previous point in time
- Automatically track and archive transactional data changes
- Roll back a transaction and its dependent transactions while the database remains online
- Undrop a table
- Recover a database to a point-in-time without a restore operation

Other than the flashback database feature, most Oracle Flashback features use the Automatic Undo Management (AUM) system to obtain metadata and historical data for transactions. They rely on undo data, which are records of the effects of individual transactions. For example, if a user runs an UPDATE statement to change a salary from 1000 to 1100, then Oracle Database stores the value 1000 in the undo data.

Undo data is persistent and survives a database shutdown. By using flashback features, you can use undo data to query past data or recover from logical damage. Besides using it in flashback features, Oracle Database uses undo data to perform these actions:

- Roll back active transactions
- Recover terminated transactions by using database or process recovery
- Provide read consistency for SQL queries

Oracle Flashback can address and rewind data that is compromised due to various human or operator errors that inadvertently or maliciously change data, cause bad installations and upgrades, and result in logical errors in applications. These problems use features such as flashback transaction, flashback drop, flashback table, and flashback database.

> **See Also:**
>
> *Oracle Database Development Guide*
>
> Performing Flashback and Database Point-in-Time Recovery, Using Flashback Database and Restore Points, and Performing Block Media Recovery in the *Oracle Database Backup and Recovery User’s Guide*
>
> *Oracle Database PL/SQL Packages and Types Reference*
>
> *Oracle Database Backup and Recovery Reference*

Oracle Flashback Query

Oracle Flashback Query (Flashback Query) provides the ability to view data as it existed in the past by using the Automatic Undo Management system to obtain metadata and historical data for transactions.

Undo data is persistent and survives a database malfunction or shutdown. The unique features of Flashback Query not only provide the ability to query previous versions of tables, they also provide a powerful mechanism to recover from erroneous operations.

Uses of Flashback Query include:

- Recovering lost data or undoing incorrect, committed changes. For example, rows that were deleted or updated can be immediately repaired even after they were committed.
- Comparing current data with the corresponding data at some time in the past. For example, by using a daily report that shows the changes in data from yesterday, it is possible to compare individual rows of table data, or find intersections or unions of sets of rows.
- Checking the state of transactional data at a particular time, such as verifying the account balance on a certain day.
- Simplifying application design by removing the need to store certain types of temporal data. By using Flashback Query, it is possible to retrieve past data directly from the database.
- Applying packaged applications, such as report generation tools, to past data.
- Providing self-service error correction for an application, enabling users to undo and correct their errors.

Oracle Flashback Version Query

Oracle Flashback Version Query is an extension to SQL that you can use to retrieve the versions of rows in a given table that existed at a specific time interval.

Oracle Flashback Version Query returns a row for each version of the row that existed in the specified time interval. For any given table, a new row version is created each time the

```
COMMIT
```

statement is executed.

Oracle Flashback Version Query is a powerful tool that database administrators (database administrators) can use to run analysis to determine the source of problems. Additionally, application developers can use Oracle Flashback Version Query to build customized applications for auditing purposes.

Oracle Flashback Transaction

Oracle Flashback Transaction backs out a transaction and its dependent transactions.

The

```
DBMS_FLASHBACK.TRANSACTION_BACKOUT()
```

procedure rolls back a transaction and its dependent transactions while the database remains online. This recovery operation uses undo data to create and execute the compensating transactions that return the affected data to its original state. You can query the

```
DBA_FLASHBACK_TRANSACTION_STATE
```

view to see whether the transaction was backed out using dependency rules or forced out by either:

- Backing out nonconflicting rows
- Applying undo SQL

Oracle Flashback Transaction increases availability during logical recovery by quickly backing out a specific transaction or set of transactions and their dependent transactions. You use one command to back out transactions while the database remains online.

Oracle Flashback Transaction Query

Oracle Flashback Transaction Query provides a mechanism to view all of the changes made to the database at the transaction level.

When used in conjunction with Oracle Flashback Version Query, it offers a fast and efficient means to recover from a human or application error. Oracle Flashback Transaction Query increases the ability to perform online diagnosis of problems in the database by returning the database user that changed the row, and performs analysis and audits on transactions.

Oracle Flashback Table

Oracle Flashback Table recovers a table to a previous point in time.

It provides a fast, online solution for recovering a table or set of tables that were changed by a human or application error. In most cases, Oracle Flashback Table alleviates the need for administrators to perform more complicated point-in-time recovery operations. The data in the original table is not lost when you use Oracle Flashback Table because you can return the table to its original state.

Oracle Flashback Drop

Although there is no easy way to recover dropped tables, indexes, constraints, or triggers, Oracle Flashback Drop provides a safety net when you are dropping objects.

When you drop a table, it is automatically placed into the Recycle Bin. The Recycle Bin is a virtual container where all dropped objects reside. You can continue to query data in a dropped table.

Restore Points

When an Oracle Flashback recovery operation is performed on the database, you must determine the point in time—identified by the system change number (SCN) or time stamp—to which you can later flash back the data.

Oracle Flashback restore points are labels that you can define to substitute for the SCN or transaction time used in Flashback Database, Flashback Table, and Oracle Recovery Manager (RMAN) operations. Furthermore, a database can be flashed back through a previous database recovery and opened with an

```
OPEN RESETLOGS
```

command by using guaranteed restore points. Guaranteed restore points allow major database changes—such as database batch jobs, upgrades, or patches—to be quickly undone by ensuring that the undo required to rewind the database is retained.

Using the restore points feature provides the following benefits:

- The ability to quickly restore to a consistent state, to a time before a planned operation that has gone awry (for example, a failed batch job, an Oracle software upgrade, or an application upgrade)
- The ability to resynchronize a snapshot standby database with the primary database
- A quick mechanism to restore a test or cloned database to its original state

Oracle Flashback Database

Oracle Flashback Database is the equivalent of a fast rewind button, quickly returning a database to a previous point in time without requiring a time consuming restore and roll forward using a backup and archived logs.

The larger the size of the database, the greater the advantage of using Oracle Flashback Database for fast point in time recovery.

Enabling Oracle Flashback Database provides the following benefits:

- Fast point in time recovery to repair logical corruptions, such as those caused by administrative error.
- Useful for iterative testing when used with Oracle restore points. A restore point can be set, database changes implemented, and test workload run to assess impact. Oracle Flashback Database can then be used to discard the changes and return the database to the original starting point, different modifications can be made, and the same test workload run a second time to have a true basis for comparing the impact of the different configuration changes.
- Data Guard uses Oracle Flashback Database to quickly reinstantiate a failed primary database as a new standby (after a failover has occurred), without requiring the failed primary to be restored from a backup.
- Flashback database operates at the CDB level or the PDB level.

Flashback Pluggable Database

You can rewind a PDB to a previous SCN. The `FLASHBACK PLUGGABLE DATABASE` command, which is available through SQL or Recovery Manager, is analogous to `FLASHBACK DATABASE` in a non-CDB.

Flashback PDB protects an individual PDB against data corruption, widespread user errors, and redo corruption. The operation does not rewind data in other PDBs in the CDB.

You can use

```
CREATE RESTORE POINT ... FOR PLUGGABLE
        DATABASE
```

to create a PDB restore point, which is only usable within a specified PDB. As with CDB restore points, PDB restore points can be normal or guaranteed. A guaranteed restore point never ages out of the control file and must be explicitly dropped. If you connect to the root, and if you do not specify the

```
FOR PLUGGABLE
        DATABASE
```

clause, then you create a CDB restore point, which is usable by all PDBs.

A special type of PDB restore point is a clean restore point, which you can only create when a PDB is closed. For PDBs with shared undo, rewinding the PDB to a clean restore point is faster than other options because it does not require restoring backups or creating a temporary database instance.

Block Media Recovery Using Flashback Logs or Physical Standby Database

After attempting to automatically repair corrupted blocks, block media recovery can optionally retrieve a more recent copy of a data block from the flashback logs to reduce recovery time.

Automatic block repair allows corrupt blocks on the primary database to be automatically repaired as soon as they are detected, by using good blocks from a physical standby database.

Furthermore, a corrupted block encountered during instance recovery does not result in instance recovery failure. The block is automatically marked as corrupt and added to the RMAN corruption list in the

```
V$DATABASE_BLOCK_CORRUPTION
```

table. You can subsequently issue the RMAN

```
RECOVER BLOCK
```

command to fix the associated block. In addition, the RMAN

```
RECOVER BLOCK
```

command restores blocks from a physical standby database, if it is available.

Flashback Data Archive

The Flashback Data Archive is stored in a tablespace and contains transactional changes to every record in a table for the duration of the record's lifetime.

The archived data can be retained for a much longer duration than the retention period offered by an undo tablespace, and used to retrieve very old data for analysis and repair.

Oracle Data Pump and Data Transport

Oracle Data Pump technology enables very high-speed movement of data and metadata from one database to another. Data Pump is used to perform the following planned maintenance activities:

- Database migration to a different platform
- Database migration to pluggable databases
- Database upgrade

The Data Pump features that enable the planned maintenance activities listed above are the following:

- Full transportable export/import to move an entire database to a different database instance
- Transportable tablespaces to move a set of tablespaces between databases

See Also:

Transporting Data

Oracle Replication Technologies for Non-Database Files

Oracle ASM Cluster File System, Oracle Database File System, and Oracle Solaris ZFS Storage Appliance Replication are the Oracle replication technologies for non-database files.

Table 3-2 Oracle Replication Technologies for Non-Database Files

Technology	Recommended Usage	Comments
Oracle ASM Cluster File System	Recommended to provide a single-node and cluster-wide file system solution integrated with Oracle ASM, Oracle Clusterware, and Oracle Enterprise Manager technologies. Provides a loosely coupled full stack replication solution when combined with Data Guard or Oracle GoldenGate.	Oracle ACFS establishes and maintains communication with the Oracle ASM instance to participate in Oracle ASM state transitions including Oracle ASM instance and disk group status updates and disk group rebalancing. Supports many database and application files, including executables, database trace files, database alert logs, application reports, BFILEs, and configuration files. Other supported files are video, audio, text, images, engineering drawings, and other general-purpose application file data. Can provide near-time consistency between database changes and file system changes when point-in-time recovery happens Can be exported and accessed by remote clients using standard NAS File Access Protocols such as NFS and CIFS.

Table 3-2 (Cont.) Oracle Replication Technologies for Non-Database Files

Technology	Recommended Usage	Comments
Oracle Database File System	Recommended for providing stronger synchronization between database and non-database systems.	Can be integrated with the database to maintain complete consistency between the database changes and the file system changes All data stored in the database and can be used with Oracle Active Data Guard to provide both disaster recovery and read-only access Can take advantage all of the Oracle database features
Oracle Solaris ZFS Storage Appliance Replication	Recommended for disaster recovery protection for non-database files, and specifically for Oracle Fusion Middleware critical files stored outside of the database.	Replicates all non-database objects, including Oracle Fusion Middleware binaries configuration Can provide near time consistency between database changes and file system changes when point-in-time recovery happens

Oracle ASM Cluster File System

Oracle ASM Cluster File System (ACFS) is a multiplatform, scalable file system, and storage management technology that extends Oracle Automatic Storage Management (Oracle ASM) functionality to support customer files maintained outside of Oracle Database.

Oracle ACFS supports many database and application files, including executables, database trace files, database alert logs, application reports, BFILEs, and configuration files. Other supported files are video, audio, text, images, engineering drawings, and other general-purpose application file data.

Oracle ACFS takes advantage of the following Oracle ASM functionality:

- Oracle ACFS dynamic file system resizing
- Maximized performance through direct access to Oracle ASM disk group storage
- Balanced distribution of Oracle ACFS across Oracle ASM disk group storage for increased I/O parallelism
- Data reliability through Oracle ASM mirroring protection mechanisms

Oracle ACFS Replication, similar to Data Guard for the database, enables replication of Oracle ACFS file systems across the network to a remote site, providing disaster recovery capability for the file system. Oracle ACFS replication captures file system changes written to disk for a primary file system and records the changes in files called replication logs. These logs are transported to the site hosting the associated standby file system where background processes read the logs and apply the changes recorded in the logs to the standby file system. After the changes recorded in a replication log are successfully applied to the standby file system, the replication log is deleted from the sites hosting the primary and standby file systems.

An additional feature of Oracle ACFS is that it offers snapshot-based replication for generic and application files, providing an HA solution for disaster recovery and Test/

Development environments. Oracle Databases stored in ACFS can leverage Oracle Multiltenant and ACFS snapshot technologies to create quick and efficient snapshot clones of pluggable databases.

Oracle Data Guard and Oracle ACFS can be combined to provide a full stack high availability solution with Data Guard protecting the database with a standby database and Oracle ACFS replicating the file system changes to the standby host. For planned outages the file system and the database remain consistent to a point in time with zero data loss.

See Also:

Oracle ACFS ASM Cluster File System: What is it and How to use it

http://www.oracle.com/goto/maa for Oracle MAA white paper “Full Stack Role Transition - Oracle ACFS and Oracle Data Guard”

Oracle Database File System

Oracle Database File System (DBFS) takes advantage of the features of the database to store files, and the strengths of the database in efficiently managing relational data, to implement a standard file system interface for files stored in the database.

With this interface, storing files in the database is no longer limited to programs specifically written to use BLOB and CLOB programmatic interfaces. Files in the database can now be transparently accessed using any operating system (OS) program that acts on files. For example, extract, transform, and load (ETL) tools can transparently store staging files in the database.

Oracle DBFS provides the following benefits:

- Full stack integration recovery and failover: By storing file system files in a database structure, it is possible to easily perform point-in-time recovery of both database objects and file system data.
- Disaster Recovery System Return on Investment (ROI): All changes to files contained in DBFS are also logged through the Oracle database redo log stream and thus can be passed to a Data Guard physical standby database. Using Oracle Active Data Guard technology, the DBFS file system can be mounted read-only using the physical standby database as the source. Changes made on the primary are propagated to the standby database and are visible once applied to the standby.
- File system backups: Because DBFS is stored in the database as database objects, standard RMAN backup and recovery functionality can be applied to file system data. Any backup, restore, or recovery operation that can be performed on a database or object within a database can also be performed against the DBFS file system.

See Also:

Database File System (DBFS)

Oracle Solaris ZFS Storage Appliance Replication

The Oracle Solaris ZFS Storage Appliance series supports snapshot-based replication of projects and shares from a source appliance to any number of target appliances manually, on a schedule, or continuously.

The Oracle Solaris ZFS Storage Appliance series supports the following use cases:

- Disaster recovery: Replication can be used to mirror an appliance for disaster recovery. In the event of a disaster that impacts the service of the primary appliance (or even an entire data center), administrators activate the service at the disaster recovery site, which takes over using the most recently replicated data. When the primary site is restored, data changed while the disaster recovery site was in service can be migrated back to the primary site, and normal service is restored. Such scenarios are fully testable before a disaster occurs.
- Data distribution: Replication can be used to distribute data (such as virtual machine images or media) to remote systems across the world in situations where clients of the target appliance would not ordinarily be able to reach the source appliance directly, or such a setup would have prohibitively high latency. One example uses this scheme for local caching to improve latency of read-only data (such as documents).
- Disk-to-disk backup: Replication can be used as a backup solution for environments in which tape backups are not feasible. Tape backup might not be feasible, for example, because the available bandwidth is insufficient or because the latency for recovery is too high.
- Data migration: Replication can be used to migrate data and configuration between Oracle Solaris ZFS Storage appliances when upgrading hardware or rebalancing storage. Shadow migration can also be used for this purpose.

The architecture of Oracle Solaris ZFS Storage Appliance also makes it an ideal platform to complement Data Guard for disaster recovery of Oracle Fusion Middleware. Oracle Fusion Middleware has a number of critical files that are stored outside of the database. These binaries, configuration data, metadata, logs and so on also require data protection to ensure availability of the Oracle Fusion Middleware. For these, the built-in replication feature of the ZFS Storage Appliance is used to move this data to a remote disaster recovery site.

Benefits of the Oracle Solaris ZFS Storage Appliance when used with Oracle Fusion Middleware include:

- Leverages remote replication for Oracle Fusion Middleware
- Provides ability to quickly create clones and snapshots of databases to increase ROI of DR sites

See Also:

Oracle ZFS Storage Appliance Software

Oracle Multitenant

Oracle Multitenant is the optimal database consolidation method. The multitenant architecture combines the best attributes of each of the previous consolidation methods without their accompanying tradeoffs.

Oracle Multitenant helps reduce IT costs by simplifying consolidation, provisioning, upgrades and more. This new architecture allows a container database (CDB) to hold many pluggable databases (PDBs). To applications, these PDBs appear as a standalone database, and no changes are required to the application in order to access the PDB. By consolidating multiple databases as PDBs into a single CDB, you are provided with the ability to manage "many as one". The flexibility remains to operate on PDBs in isolation should your business require it.

Oracle Multitenant is fully compliant with and takes direct advantage of high availability features such as Oracle Real Application Clusters, Oracle Data Guard, and Oracle GoldenGate, just like any non-container database (non-CDB), meaning it can be used in any of the Oracle MAA reference architectures. Grouping multiple PDBs with the same high availability requirements into the same CDB ensures that all of those PDBs and their applications are managed and protected with the same technologies and configurations.

Benefits of Using Oracle Multitenant

- High consolidation density - Many PDBs can be stored in a single CDB. These PDBs share background processes and memory structures letting you run more PDBs than you would non-CDBs, because the overhead for each non-CDB is removed or reduced. You can store up to 4095 PDBs in a CDB. Each PDB can also have a different characterset from other PDBs within the same CDB, as long as the CDB root character set is a superset of all of the PDBs’ character sets. Logical standby databases also support this mix of character sets to allow rolling upgrades with a transient logical standby database.
- Online provisioning operations, including clones, refreshable clones, and PDB relocation - A PDB can be unplugged from one CDB and plugged into another. A PDB can also be cloned either into the same CDB or into a different CDB. Cloning can be used to create a "gold image" or seed database for DBaaS or SaaS environments. This PDB can then be rapidly cloned to easily set up database environments for new customers.
  - Near Zero Downtime PDB Relocation – This feature significantly reduces the downtime of relocating a PDB from one CDB to another by using clone functionality. The source PDB remains open and functional while the relocation takes place. The application outage is reduced to a very short window while the source PDB is brought to a consistent state, and the destination PDB is synchronized and brought online. This functionality also takes advantage of another new feature, Listener Redirects, which allows you to keep the same connect descriptor for applications and connect to the destination PDB even after it has been relocated.
  - Online provisioning and cloning – Clones of PDBs can be created without requiring the source PDB to be placed in read only-mode. The source PDB can be left in read-write mode and accessible to applications for the duration of the clone operation.
  - Refreshable Clone PDB – Clones of PDBs can be created in such a way as to be refreshed with changes with changes made to the source PDB applied either automatically at set intervals or manually. For a clone to be refreshable it must remain in read-only mode. The clone can be converted into an ordinary PDB by opening it read-write. Refreshable clones are well suited to be used as test masters for Exadata storage snapshots.

- New patching and upgrade options -When you upgrade or patch a CBD, all of the PDBs in that container are also upgraded or patched. If you need isolation, you can unplug a PDB and plug it into a CDB at a later version.
- Database backup and recovery - By consolidating multiple databases as PDBs, operations such as backup and disaster recovery are performed at the container level. Oracle Multitenant also provides the flexibility to backup and restore individual PDBs with no impact to other running PDBs in the same CDB.
- Operation with Oracle Data Guard - Data Guard configurations are maintained at the CDB level. When a Data Guard role transition (either failover or switchover) is performed, all PDBs are transitioned to the new primary database. There is no need to create or manage multiple Data Guard configurations for each PDB as would be required for single databases. Existing tools such as Data Guard Standby First Patching and Data Guard Transient Logical Rolling Upgrade can still be used to reduce downtime and are performed at the container level, so all PDBs will be maintained in a single operation.
  - PDB Migration with Data Guard Broker – The Data Guard broker has been enhanced to provide automation for migrating PDBs from one CDB, either the primary database or the standby database, to another CDB. This can be used for straight migration of a PDB from one CDB to another running at either at the same version or a CDB running at a higher version to start the upgrade process. This automation can also be used to affect a single PDB failover by using the PDBs files at a standby database to plug into a different CDB at the same version.
  - Subset Standby - A subset standby enables users of Oracle Multitenant to designate a subset of the PDBs in a CDB for replication to a standby database. This provides a finer granularity of designating which standby databases will contain which PDBs.
- Operation with Oracle GoldenGate - All of functionality provided by Oracle GoldenGate also exists for Oracle Multitenant. GoldenGate also provides the flexibility to operate at the PDB level, allowing replication to occur for a subset of the PDBs in a CDB. GoldenGate can be used for minimal to zero downtime upgrades either at the CDB level or at an individual PDB level.
- Resource management - Just as Oracle Resource Manager can control resource utilization between single databases, it can also control resource utilization of individual PDBs in a container. This can ensure that a single PDB does not access more than its assigned share of system resources. You can specify guaranteed minimums and maximums for SGA, buffer cache, shared pool, and PGA memory at the PDB limit.
- Operation with Oracle Flashback Database - If fast point-in-time recovery is required, the initial release of Oracle Multitenant enables using Flashback Database at the CDB level. Oracle Multitenant enables Flashback Database to be used on an individual PDB without impacting the availability of other PDBs. Flashback Database can performed at the CDB level which will flashback all of the PDBs in the container. Individual PDBs can be flashed back using the Flashback Pluggable Database feature. When flashing back an individual PDB all other PDBs remain unaffected.
- Data Guard Broker PDB Migration or Failover - In multitenant broker configurations, you may need to move a Production PDB from one container database to another container database that resides on the same system. You may also need to failover a PDB from a Data Guard Standby database to a new production container database when the production PDB has failed but the

container database and all other PDBs function normally. Using the new Data Guard Broker command, `MIGRATE PLUGGABLE DATABASE`, you can easily move a single PDB from one container database to another, or failover a single PDB from a Data Guard standby to a new production container database. (new in Oracle Database 12c Release 2)

See Also:

Oracle Multitenant Administrator's Guide

Best Practices for Oracle Database Consolidation: A Guide for Implementation Including MAA Reference Architectures

Oracle Sharding

Oracle Sharding is a scalability and availability feature for applications explicitly designed to run on a sharded database.

Oracle sharding enables distribution and replication of data across a pool of Oracle databases that share no hardware or software. The pool of databases is presented to the application as a single logical database. Applications elastically scale (data, transactions, and users) to any level, on any platform, simply by adding additional databases (shards) to the pool. Scaling up to 1000 shards is supported.

Oracle Sharding provides superior run-time performance and simpler life-cycle management compared to home-grown deployments that use a similar approach to scalability. It also provides the advantages of an enterprise DBMS, including relational schema, SQL, and other programmatic interfaces, support for complex data types, online schema changes, multi-core scalability, advanced security, compression, high-availability, ACID properties, consistent reads, developer agility with JSON, and much more.

See Also:

Using Oracle Sharding

Oracle Restart

Oracle Restart enhances the availability of a single-instance (nonclustered) Oracle database and its components.

Oracle Restart is used in single-instance environments only. For Oracle Real Application Clusters (Oracle RAC) environments, the functionality to automatically restart components is provided by Oracle Clusterware.

If you install Oracle Restart, it automatically restarts the database, the listener, and other Oracle components after a hardware or software failure or whenever the database's host computer restarts. It also ensures that the Oracle components are restarted in the proper order, in accordance with component dependencies.

Oracle Restart periodically monitors the health of components—such as SQL*Plus, the Listener Control utility (LSNRCTL), ASMCMD, and Oracle Data Guard—that are integrated with Oracle Restart. If the health check fails for a component, Oracle Restart shuts down and restarts the component.

Oracle Restart runs out of the Oracle Grid Infrastructure home, which you install separately from Oracle Database homes.

Integrated client failover applications depend on role based services and Fast Application Notification events, managed by Oracle clusterware, to alert the application to failures. Single instance databases must have Oracle Restart to achieve integrated client failover.

> **See Also:**
>
> *Oracle Database Administrator’s Guide* for information about installing and configuring the Oracle Restart feature

Oracle Site Guard

Oracle Site Guard is a disaster-recovery solution that enables administrators to automate complete site switchover or failover.

Oracle Site Guard orchestrates and automates the coordinated failover of Oracle Fusion Middleware, Oracle Fusion Applications, and Oracle Databases. It is also extensible to include other data center software components.

Oracle Site Guard integrates with underlying replication mechanisms that synchronize primary and standby environments and protect mission critical data. It comes with a built-in support for Oracle Data Guard for Oracle database, and Oracle Sun ZFS. Oracle Site Guard can also support other storage replication technologies.

> **See Also:**
>
> *Oracle Enterprise Manager Oracle Site Guard Administrator's Guide*

Online Reorganization and Redefinition

One way to enhance availability and manageability is to allow user access to the database during a data reorganization operation.

The Online Reorganization and Redefinition feature in Oracle Database offers administrators significant flexibility to modify the physical attributes of a table and transform both data and table structure while allowing user access to the database. This capability improves data availability, query performance, response time, and disk space usage. All of these are important in a mission-critical environment and make the application upgrade process easier, safer, and faster.

Use Oracle Database online maintenance features to significantly reduce (or eliminate) the application downtime required to make changes to an application's database objects

See Also:

Redefining Tables Online in *Oracle Database Administrator’s Guide*

Zero Data Loss Recovery Appliance

The cloud-scale Zero Data Loss Recovery Appliance, commonly known as Recovery Appliance, is an engineered system designed to dramatically reduce data loss and backup overhead for all Oracle databases in the enterprise.

Integrated with Recovery Manager (RMAN), the Recovery Appliance enables a centralized, incremental-forever backup strategy for large numbers of databases, using cloud-scale, fault-tolerant hardware and storage. The Recovery Appliance continuously validates backups for recoverability.

Recovery Appliance is the MAA-preferred backup and recovery appliance because:

- Elimination of data loss when restoring from Recovery Appliance
- Minimal backup overhead
- Improved end-to-end data protection visibility
- Cloud-scale protection
- Integrates very well with all MAA reference architectures including Oracle Sharding tier

See Also:

Zero Data Loss Recovery Appliance Documentation

Fleet Patching and Provisioning

Fleet Patching and Provisioning maintains a space-efficient repository of software, more precisely "gold images," which are standardized software homes that can be provisioned to any number of target machines.

Any number of homes can be provisioned from a given gold image, and Fleet Patching and Provisioning maintains lineage information so that the provenance of deployed software is always known. Gold images can be organized into series, allowing you to create groupings that track the evolution of a release, with different series for different tailored solutions such as Oracle Database patch bundles for specific applications. A notification system informs interested parties when a new image is available in a given series. Fleet Patching and Provisioning is a feature of Oracle Grid Infrastructure. The components that form the Fleet Patching and Provisioning Server are managed automatically by Oracle Grid Infrastructure.

Fleet Patching and Provisioning can provision databases, clusterware, middleware, and custom software. Fleet Patching and Provisioning offers additional features for creating, configuring, patching and upgrading Oracle Grid Infrastructure and Oracle Database deployments. These capabilities simplify maintenance, reducing its risk and impact, and provide a roll-back option if changes need to be backed out. Additional capabilities include provisioning clusters and databases onto base machines, and simple capacity on demand by growing and shrinking clusters and Oracle RAC databases. All of these operations are performed with single commands which replace the numerous manual steps otherwise required. All commands and their outcomes are recorded in an audit log. All workflows allow customization to support the unique requirements of any environment.

The key benefits of Fleet Patching and Provisioning are:

- Enables and enforces standardization
- Simplifies provisioning, patching and upgrading
- Minimizes the impact and risk of maintenance
- Increases automation and reduces touch points
- Supports large scale deployments

See Also:

Fleet Patching and Provisioning and Maintenance in *Oracle Clusterware Administration and Deployment Guide*

Oracle Fleet Patching and Provisioning (FPP) Introduction and Technical Overview

Edition-Based Redefinition

Planned application changes may include changes to data, schemas, and programs. The primary objective of these changes is to improve performance, manageability, and functionality. An example is an application upgrade.

Edition-based redefinition (EBR) lets you upgrade the database component of an application while it is in use, thereby minimizing or eliminating downtime. To upgrade an application while it is in use, you must copy the database objects that comprise the database component of the application and redefine the copied objects in isolation. Your changes do not affect users of the application; they can continue to run the unchanged application. When you are sure that your changes are correct, you make the upgraded application available to all users.

> **See Also:**
>
> Using Edition-Based Redefinition in *Oracle Database Development Guide*

4

Oracle Database High Availability Solutions for Unplanned Downtime

Oracle Database offers an integrated suite of high availability solutions that increase availability.

These solutions also **eliminate or minimize** both planned and unplanned downtime, and help enterprises maintain business continuity 24 hours a day, 7 days a week. However, Oracle's high availability solutions not only go beyond reducing downtime, but also help to improve overall performance, scalability, and manageability.

Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Various Oracle MAA high availability solutions for unplanned downtime are described here in an easy to navigate matrix.

The following table shows how the features discussed in the referenced (hyperlinked) sections can be used to address various causes of unplanned downtime. Where several Oracle solutions are listed, the MAA recommended solution is indicated in the Oracle MAA Solution column.

Table 4-1 Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
Site failures	Oracle Data Guard and Enabling Continuous Service for Applications (MAA recommended)	• Integrated client and application failover • Fastest and simplest database replication • Supports all data types • Zero data loss by eliminating propagation delay • Oracle Active Data Guard – Supports read-only services and DML on global temporary tables and sequences to off-load more work from the primary – Allows small updates to be redirected to the primary enabling read-mostly reports to be offloaded to standby • Database In-Memory support
	Oracle GoldenGate	• Flexible logical replication solution (target is open read/write) • Active-active high availability (with conflict resolution) • Heterogeneous platform and heterogeneous database support • Potential zero downtime with custom application failover

Table 4-1 (Cont.) Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
	Recovery Manager, Zero Data Loss Recovery Appliance and Oracle Secure Backup	• Fully managed database recovery and integration with Oracle Secure Backup • Recovery Appliance – provides end-to-end data protection for backups – reduces data loss for database restores – Non-real-time recovery
Instance or computer failures	Oracle Real Application Clusters and Oracle Clusterware and Enabling Continuous Service for Applications (MAA recommended)	• Integrated client and application failover • Automatic recovery of failed nodes and instances • Lowest application brownout with Oracle Real Application Clusters
	Oracle RAC One Node and Enabling Continuous Service for Applications	• Integrated client and application failover • Online database relocation migrates connections and instances to another node • Better database availability than traditional cold failover solutions
	Oracle Data Guard and Enabling Continuous Service for Applications	• Integrated client and application failover • Fastest and simplest database replication • Supports all data types • Zero data loss by eliminating propagation delay • Oracle Active Data Guard – Supports read-only services and DML on global temporary tables and sequences to off-load more work from the primary – Allows small updates to be redirected to the primary enabling read-mostly reports to be offloaded to standby • Database In-Memory support
	Oracle GoldenGate	• Flexible logical replication solution (target is open read/write) • Active-Active high availability (with conflict resolution) • Heterogeneous platform and heterogeneous database support • Potential zero downtime with custom application failover
Storage failures	Oracle Automatic Storage Management (MAA recommended)	Mirroring and online automatic rebalancing places redundant copies of the data in separate failure groups.
	Oracle Data Guard (MAA recommended)	• Integrated client and application failover • Fastest and simplest database replication • Supports all data types • Zero data loss by eliminating propagation delay • Oracle Active Data Guard supports read-only services and DML on global temporary tables and sequences to off-load more work from the primary • Database In-Memory support

Table 4-1 (Cont.) Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
	Recovery Manager with Fast Recovery Area, and Zero Data Loss Recovery Appliance (MAA recommended)	Fully managed database recovery and managed disk and tape backups
	Oracle GoldenGate	• Flexible logical replication solution (target is open read/write) • Active-active high availability (with conflict resolution) • Heterogeneous platform and heterogeneous database support • Potential zero downtime with custom application failover
Data corruption	Corruption Prevention, Detection, and Repair (MAA recommended) Database initialization settings such as `DB_BLOCK_CHECKING`, `DB_BLOCK_CHECKSUM`, and `DB_LOST_WRITE_PROTECT`	Different levels of data and redo block corruption prevention and detection at the database level

Table 4-1 (Cont.) Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
Data corruption	Oracle Data Guard (MAA recommended) Oracle Active Data Guard Automatic Block Repair `DB_LOST_WRITE_PROTECT` initialization parameter	• In a **Data Guard** configuration with an **Oracle Active Data Guard** standby – Physical block corruptions detected by Oracle at a primary database are automatically repaired using a good copy of the block retrieved from the standby, and vice versa – The repair is transparent to the user and application, and data corruptions can definitely be isolated • With **MAA recommended initialization settings,** Oracle Active Data Guard and Oracle Exadata Database Machine, achieve most comprehensive full stack corruption protection. • With **`DB_LOST_WRITE_PROTECT`** enabled – A lost write that occurred on the primary database is detected either by the physical standby database or during media recovery of the primary database, recovery is stopped to preserve the consistency of the database – Failing over to the standby database using Data Guard will result in some data loss – Data Guard Broker's `PrimaryLostWrite` property supports SHUTDOWN and CONTINUE, plus FAILOVER and FORCEFAILOVER options, when lost writes are detected on the primary database. See *Oracle Data Guard Broker* – `DB_LOST_WRITE_PROTECT` initialization parameter provides lost write detection • **Shadow lost write protection** detects a lost write before it can result in major data corruption. You can enable shadow lost write protection for a database, a tablespace, or a data file without requiring an Oracle Data Guard standby database. Note the impact on your workload may vary.

Table 4-1 (Cont.) Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
	Dbverify, Analyze, Data Recovery Advisor and Recovery Manager, Zero Data Loss Recovery Appliance, and ASM Scrub with Fast Recovery Area (MAA recommended)	These tools allow the administrator to execute manual checks to help detect and potentially repair from various data corruptions. • **Dbverify** and **Analyze** conduct physical block and logical intra-block checks. Analyze can conduct inter-object consistency checks. • **Data Recovery Advisor** automatically detects data corruptions and recommends the best recovery plan. • **RMAN** operations can – Conduct both physical and inter-block logical checks – Run online block-media recovery using flashback logs, backups, or the standby database to help recover from physical block corruptions • **Recovery Appliance** – Does periodic backup validation that helps ensure that your backups are valid – Allows you to input your recovery window requirements, and alerts you when those SLAs cannot be met with your existing backups managed by Recovery Appliance • **ASM Scrub** detects and attempts to repair physical and logical data corruptions with the ASM pair in normal and high redundancy disks groups.
Data corruption	Oracle Exadata Database Machine and Oracle Automatic Storage Management (MAA recommended) DIX + T10 DIF Extensions (MAA recommended where applicable)	• If Oracle ASM detects a corruption and has a good mirror, ASM returns the good block and repairs the corruption during a subsequent write I/O. • Exadata provides implicit HARD enabled checks to prevent data corruptions caused by bad or misdirected storage I/O. • Exadata provides automatic HARD disk scrub and repair. Detects and fixes bad sectors. • DIX +T10 DIF Extensions provides end to end data integrity for reads and writes through a checksum validation from a vendor's host adapter to the storage device
	Oracle GoldenGate	• Flexible logical replication solution (target is open read/write). Logical replica can be used as a failover target if partner replica is corrupted. • Active-active high availability (with conflict resolution) • Heterogeneous platform and heterogeneous database support
Human errors	Oracle security features (MAA recommended)	Restrict access to prevent human errors
	Oracle Flashback Technology (MAA recommended)	• Fine-grained error investigation of incorrect results • Fine-grained and database-wide or pluggable database rewind and recovery capabilities

Table 4-1 (Cont.) Outage Types and Oracle High Availability Solutions for Unplanned Downtime

Outage Scope	Oracle MAA Solution	Benefits
Delays or slow downs	Oracle Database and Oracle Enterprise Manager Oracle Data Guard (MAA recommended) and Enabling Continuous Service for Applications	• **Oracle Database** automatically monitors for instance and database delays or cluster slow downs and attempts to remove blocking processes or instances to prevent prolonged delays or unnecessary node evictions. • **Oracle Enterprise Manager** or a customized application heartbeat can be configured to detect application or response time slowdown and react to these SLA breaches. For example, you can configure the Enterprise Manager Beacon to monitor and detect application response times. Then, after a certain threshold expires, Enterprise Manager can call the Data Guard `DBMS_DG.INITIATE_FS_FAILOVER` PL/SQL procedure to initiate a failover. See the section about "Managing Fast-Start Failover" in *Oracle Data Guard Broker*. • Database In-Memory support
File system data	Oracle Replication Technologies for Non-Database Files	Enables full stack failover that includes non-database files

Managing Unplanned Outages for MAA Reference Architectures and Multitenant Architectures

High availability solutions in each of the MAA service-level tiers for the MAA reference architectures and multitenant architectures are described in an easy to navigate matrix.

If you are managing many databases in DBaaS, we recommend using the MAA tiers and Oracle Multitenant as described in Oracle MAA Reference Architectures.

The following table identifies various unplanned outages that can impact a database in a multitenant architecture. It also identifies the Oracle high availability solution to address that outage that is available in each of the MAA reference architectures.

Table 4-2 Unplanned Outage Matrix for MAA Reference Architectures and Multitenant Architectures

Event	Solutions by MAA Architecture	Recovery Window (RTO)	Data Loss (RPO)
Instance Failure	BRONZE: Oracle Restart	Minutes if instance can restart	Zero

Table 4-2 (Cont.) Unplanned Outage Matrix for MAA Reference Architectures and Multitenant Architectures

Event	Solutions by MAA Architecture	Recovery Window (RTO)	Data Loss (RPO)
	SILVER: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware) or Oracle RAC One Node, and Enabling Continuous Service for Applications	Seconds with Oracle RAC, minutes with Oracle RAC One Node	Zero
	GOLD: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware and Enabling Continuous Service for Applications	Seconds	Zero
	PLATINUM: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware) and Enabling Continuous Service for Applications	Zero Application Outage	Zero
Permanent Node Failure (but storage available)	BRONZE: Restore and recover	Hours to Day	Zero
	SILVER: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware) and Enabling Continuous Service for Applications	Seconds	Zero
	SILVER: Oracle RAC One Node and Enabling Continuous Service for Applications	Minutes	Zero
	GOLD: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware) and Enabling Continuous Service for Applications	Seconds	Zero

Table 4-2 (Cont.) Unplanned Outage Matrix for MAA Reference Architectures and Multitenant Architectures

Event	Solutions by MAA Architecture	Recovery Window (RTO)	Data Loss (RPO)
	PLATINUM: Oracle RAC (see Oracle Real Application Clusters and Oracle Clusterware) and Enabling Continuous Service for Applications	Seconds	Zero
Storage Failure	ALL: Oracle Automatic Storage Management	Zero downtime	Zero
Data corruptions	BRONZE/SILVER: Basic protection Some corruptions require recover restore and recovery of pluggable database (PDB), entire multitenant container database (CDB) or non-container database (non-CDB)	Hour to Days	• Since last backup if unrecoverable • Zero or Near Zero with Recovery Appliance
	GOLD: 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
	PLATINUM: Comprehensive corruption protection and Auto Block Repair with Oracle Active Data Guard Oracle GoldenGate replica with custom application failover	• Zero with auto block repair • Zero with Oracle GoldenGate replica	Zero when using Active Data Guard Fast-Start Failover and Oracle GoldenGate

Table 4-2 (Cont.) Unplanned Outage Matrix for MAA Reference Architectures and Multitenant Architectures

Event	Solutions by MAA Architecture	Recovery Window (RTO)	Data Loss (RPO)
Human error	ALL: Logical failures resolved by flashback drop, flashback table, flashback transaction, flashback query flashback pluggable database, 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 pluggable database	Dependent on detection time	Dependent on logical failure
Database unusable, system, site or storage failures, wide spread corruptions or disasters	BRONZE/SILVER: Restore and recover	Hours to Days	• Since last database and archive backup • Zero or near zero with Recovery Appliance
	GOLD: Active Data Guard Fast-Start Failover and Enabling Continuous Service for Applications	Seconds	Zero to Near Zero
	PLATINUM: Oracle GoldenGate replica with custom application failover	Zero	Zero when using Active Data Guard Fast-Start Failover and Oracle GoldenGate
Performance Degradation	ALL: Oracle Enterprise Manager for monitoring and detection, Database Resource Management for Resource Limits and ongoing Performance Tuning	No downtime but degraded service	Zero

5
Oracle Database High Availability Solutions for Planned Downtime

Planned downtime can be just as disruptive to operations as unplanned downtime. This is especially true for global enterprises that must support users in multiple time zones, or for those that must provide Internet access to customers 24 hours a day, 7 days a week.

See the following topics to learn about keeping your database highly available during planned downtime.

Oracle High Availability Solutions for Planned Maintenance

Oracle provides high availability solutions for all planned maintenance.

The following table describes the various Oracle high availability solutions and their projected downtime for various maintenance activities.

Table 5-1 Oracle High Availability Solutions for Planned Maintenance

Maintenance Event	High Availability Solutions with Target Outage Time
Dynamic and Online Resource Provisioning, or Online reorganization and redefinition	**Zero application and database downtime** for • Changing initialization parameters dynamically • Renaming and relocating datafiles online • Automatic memory management tuning • Online reorganization and redefinition (managing tables and managing indexes) See the Oracle Database Administrator Guide, Oracle Database Reference Guide (to evaluate which parameters on dynamic), and Online Data Reorganization and Redefinition
Operating system software or hardware updates and patches	**Zero database downtime** with Oracle RAC and Oracle RAC One Node Rolling or Fleet Patching and Provisioning **Zero application downtime** with Application Checklist for Continuous Service for MAA Solutions **Seconds to minutes database downtime** with Standby-First Patch Apply and subsequent Data Guard Switchover
Oracle interim or diagnostic software updates or patches	**Zero downtime** with Online Patching **Zero database downtime** with Oracle RAC and Oracle RAC One Node. **Zero application downtime** with Application Checklist for Continuous Service for MAA Solutions

Table 5-1 (Cont.) Oracle High Availability Solutions for Planned Maintenance

Maintenance Event	High Availability Solutions with Target Outage Time
Oracle Database or Grid Infrastructure quarterly updates under the Critical Patch Update (CPU) program, or Oracle Grid Infrastructure release upgrades	**Zero database downtime** with Oracle RAC and Oracle RAC One Node Rolling. **Zero application downtime** with Application Checklist for Continuous Service for MAA Solutions **Seconds to minutes downtime** with Standby-First Patch Apply and subsequent Data Guard Switchover
Oracle Database Release Upgrade (for example, Oracle Database 11g to 12.2 or 12.2 to 19c)	**Seconds to minutes downtime** with Data Guard transient logical or DBMS_ROLLING solution **Zero downtime** with Oracle GoldenGate See Automated Database Upgrades using Oracle Active Data Guard and DBMS_ROLLING for 12.2 and higher database releases or Database Rolling Upgrade using Data Guard for older releases.
Exadata storage or Exadata switch software updates	**Zero downtime** using Exadata `patchmgr` See Updating Exadata Software
Database Server or Oracle RAC cluster changes (add node, drop node, adjust CPU or memory size of the database server)	Some hardware changes like adjusting CPU can be done online without restarting the database server. Refer to the hardware specific documentation. If the change is not online, then **Zero database downtime** with Oracle RAC and Oracle RAC One Node Rolling. **Zero application downtime** with Application Checklist for Continuous Service for MAA Solutions **Seconds to minutes downtime** with Standby-First Patch Apply and subsequent Data Guard Switchover
Application upgrades	**Zero downtime** with Edition Based Redefinition **Zero downtime** with Oracle GoldenGate See Edition Based Redefinition and Oracle GoldenGate documentation

High Availability Solutions for Migration

Oracle MAA recommends several solutions for reducing downtime due to database migration.

The following table describes the high availability solutions for migration at a high level.

Table 5-2 High Availability Solutions for Migration

Maintenance Event	High Availability Solutions with Target Outage Time
Migrate to Oracle Cloud or any Oracle Exadata Database Machine system (Oracle Cloud includes Exadata Cloud Service (ExaCS), Exadata Cloud@Customer (ExaCC) ,Autonomous Database (ADB) or Database Cloud Service (DBCS)	To migrate to an Oracle Exadata Database Machine on-premises or any Oracle Cloud service, use the Zero Downtime Migration tool, which provides • Physical migration with RMAN backup and restore, with an optional low downtime option using Data Guard • Logical migration with Oracle Data Pump, with an optional low downtime option using Oracle GoldenGate To migrate to Oracle Autonomous Database, use the Oracle Cloud Infrastructure Database Migration service (or the Zero Downtime Migration tool), which provide • Offline migration with Data Pump • Online migration with Data Pump and Oracle GoldenGate
Migrate the database to a different server or platform	**Seconds to minutes downtime** with Oracle Data Guard for certain platform combinations **Zero downtime** with Oracle GoldenGate Data Guard always supports primary and standby combinations on the same platform. For heterogeneous platforms, Refer to Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration (Doc ID 413484.1)
Migrate database to an incompatible character set	**Zero downtime** with Oracle GoldenGate See Character Set Migration
Migrate to pluggable databases to another container database	**Seconds to minutes downtime** with Pluggable Database Relocate (PDB Relocate) See Relocating a PDB
Migrate to new storage	**Zero Downtime** with Oracle Automatic Storage Management if storage is compatible with Oracle Data Guard for certain platform combinations **Zero Downtime** with Oracle GoldenGate
Migrate database from a single-instance system to an Oracle RAC cluster	**Zero Downtime** with Oracle RAC when applicable. See Adding Oracle RAC to Nodes with Oracle Clusterware Installed **Seconds to minutes downtime** with Oracle Data Guard for certain platform combinations **Zero Downtime** with Oracle GoldenGate

6 Enabling Continuous Service for Applications

Applications achieve continuous service easily when the underlying network, systems, and databases are always available.

To achieve continuous service in the face of unplanned outages and planned maintenance activities can be challenging. An MAA database architecture and its configuration and operational best practices is built upon redundancy and its ability to tolerate, prevent, and at times auto-repair failures.

However, applications can incur downtime whenever a failure hits a database instance, a database node, or the entire cluster or data center. Similarly, some planned maintenance activities may require restarting a database instance, a database node, or an entire database server to be restarted.

In all cases, following a simple checklist, your applications can incur zero or very little downtime whenever the database service that the application is connected to can be moved to another Oracle RAC instance or to another database.

To achieve continuous application uptime during Oracle RAC switchover or failover events, follow these application configuration best practices:

- Use non-default Oracle Clusterware-managed services to connect your application.
- Use recommended connection strings with built-in timeouts, retries, and delays, so that incoming connections do not see errors during outages.
- Configure your connections with Fast Application Notification.
- Drain and relocate services before any planned maintenance requiring an Oracle RAC instance restart.

  Software updates to Exadata Database Host or Exadata Database Guest automatically drain and relocate services. Oracle Cloud and Fleet Patching and Provisioning (FPP) drain and relocate services automatically for Oracle Database, Grid Infrastructure, and Exadata software updates.
- Leverage Application Continuity or Transparent Application Continuity to replay in-flight uncommitted transactions transparently after failures.

Depending on the planned maintenance event, Oracle attempts to automatically drain and relocate application services before restarting any Oracle RAC instance. For OLTP applications, draining and relocating services works very well and results in zero application downtime.

Some applications such as long running batch jobs or reports may not be able to drain and relocate gracefully or within the maximum drain timeout. For those applications, Oracle recommends scheduling the software planned maintenance window that contains Oracle RAC rolling activities to exclude these types of activities by picking a window that will not conflict with these activities, or stopping these activities before the planned maintenance window. For example, you can reschedule a planned maintenance window outside your batch windows, or stop challenging batch jobs or reports before a planned maintenance window.

The following table outlines planned maintenance events that will incur Oracle RAC instance rolling restart and the relevant service drain timeout variables that may impact your application.

Table 6-1 Drain Timeout Variables for Planned Maintenance Events

Planned Maintenance Event	Drain Timeout Variables
Exadata Database Host (Dom0) software changes	Exadata Host handles operating system (OS) shutdown with maximum timeout of 10 minutes. OS shutdown calls an `rhphelper`, which has the following drain timeout settings: • `DRAIN_TIMEOUT`: value used for services that do not have a `drain_timeout` defined. Default 180 • `MAX_DRAIN_TIMEOUT`: overrides any higher `drain_timeout` value defined for a given service. Default 300 Each Clusterware-managed service is also controlled by a `drain_timeout` attribute that can be lower than the above values. See also: Using RHPhelper to Minimize Downtime During Planned Maintenance on Exadata (Doc ID 2385790.1)
Exadata Database Guest (DomU) software changes	Exadata `patchmgr` and `dbnodeupdate` software programs call `rhphelper`, which has the following drain timeout settings: `DRAIN_TIMEOUT`: value used for services that do not have a `drain_timeout` defined. Default 180 `MAX_DRAIN_TIMEOUT`: overrides any higher `drain_timeout` value defined for a given service. Default 300 Each Clusterware-managed service is also controlled by a `drain_timeout` attribute that can be lower than the above values. See also: Using RHPhelper to Minimize Downtime During Planned Maintenance on Exadata (Doc ID 2385790.1)
Oracle Grid Infrastructure (GI) software changes or upgrade	The recommend steps are described in Graceful Application Switchover in RAC with No Application Interruption (Doc ID 1593712.1). Example: `srvctl stop instance -o immediate -drain_timeout 600 -failover -force` Each Clusterware-managed service is also controlled by a `drain_timeout` attribute that can be lower than the above values.

Table 6-1 (Cont.) Drain Timeout Variables for Planned Maintenance Events

Planned Maintenance Event	Drain Timeout Variables
Oracle Database Software changes	The recommend steps are described in Graceful Application Switchover in RAC with No Application Interruption (Doc ID 1593712.1). Example: `srvctl stop instance -o immediate -drain_timeout 600 -failover -force` Each Clusterware-managed service is also controlled by a `drain_timeout` attribute that can be lower than the above values.

For more information, see Application Checklist for Continuous Service for MAA Solutions for recommendations to experience application-level service uptime similar to that of the database uptime. Oracle recommends testing your application readiness by following the recommendations in Validating Application Failover Readiness (Doc ID 2758734.1).

7 Operational Prerequisites to Maximizing Availability

Use the following operational best practices to provide a successful MAA implementation.

Understand Availability and Performance SLAs

Understand and document your high availability and performance service-level agreements (SLAs):

- Understand the attributes of High Availability and various causes of downtime as described in Overview of High Availability.
- Get agreement from line of business, upper management, and technical teams on HA and performance service level agreements as described in High Availability Requirements, and A Methodology for Documenting High Availability Requirements.

Implement and Validate a High Availability Architecture That Meets Your SLAs

When you have agreement on your high availability and performance service level requirements:

- **Map** the requirements to one of the Oracle MAA standard and validated MAA reference architectures, as described in High Availability and Data Protection – Getting From Requirements to Architecture
- **Evaluate** the outage and planned maintenance matrices relevant to your referenced architecture in Oracle Database High Availability Solutions for Unplanned Downtime and Oracle Database High Availability Solutions for Planned Downtime
- **Learn** about the database features required to implement your MAA architecture in High Availability Architectures

Establish Test Practices and Environment

You must **validate** or **automate** the following to ensure that your target high availability SLAs are met:

- All software update and upgrade maintenance events
- All repair operations, including those for various types of unplanned outages
- Backup, restore, and recovery operations

If you use Oracle Data Guard for disaster recovery and data protection, Oracle recommends that you:

- Perform periodic switchover operations, or conduct full application and database failover tests
- Validate end-to-end role transition procedures by performing application and Data Guard switchovers periodically

A good test environment and proper test practices are essential prerequisites to achieving the highest stability and availability in your production environment. By validating every change in your test environment thoroughly, you can proactively detect, prevent, and avoid problems before applying the same change on your production systems.

These practices involve the following:

Configuring the Test System and QA Environments

The test system should be a replica of the production MAA environment (for example, using the MAA Gold reference architecture.) There will be trade offs if the test system is not identical to the MAA service-level driven standard reference architecture that you plan to implement. It's recommended that you execute functional, performance, and availability tests with a workload that mimics production. Evaluate if availability and performance SLAs are maintained after each change, and ensure that clear fallback or repair procedures are in place if things go awry, while applying the change on the production environment.

With a properly configured test system, many problems can be avoided, because changes are validated with an equivalent production and standby database configuration containing a full data set and using a workload framework to mimic production (for example, using Oracle Real Application Testing.)

Do not try to reduce costs by eliminating the test system, because that decision ultimately affects the stability and the availability of your production applications. Using only a subset of system resources for testing and QA has the tradeoffs shown in the following table, which is an example of the MAA Gold reference architecture.

Table 7-1 Tradeoffs for Different Test and QA Environments

Test Environment	Benefits and Tradeoffs
Full Replica of Production and Standby Systems	**Validate:** • All software updates and upgrades • All functional tests • Full performance at production scale • Full high availability
Full Replica of Production Systems	**Validate:** • All software updates and upgrades • All functional tests • Full performance at production scale • Full high availability minus the standby system **Cannot Validate:** • Functional tests, performance at scale, high availability, and disaster recovery on standby database

Table 7-1 (Cont.) Tradeoffs for Different Test and QA Environments

Test Environment	Benefits and Tradeoffs
Standby System	**Validate:** • Most software update changes • All functional tests • Full performance--if using Data Guard Snapshot Standby, but this can extend recovery time if a failover is required • Role transition • Resource management and scheduling--required if standby and test databases exist on the same system
Shared System Resource	**Validate:** • Most software update changes • All functional tests This environment may be suitable for performance testing if enough system resources can be allocated to mimic production. Typically, however, the environment includes a subset of production system resources, compromising performance validation. Resource management and scheduling is required.
Smaller or Subset of the system resources	**Validate:** • All software update changes • All functional tests • Limited full-scale high availability evaluations **Cannot Validate:** • Performance testing at production scale
Different hardware or platform system resources but same operating system	**Validate:** • Some software update changes • Limited firmware patching test • All functional tests unless limited by new hardware features • Limited production scale performance tests • Limited full-scale high availability evaluations

See Also:

Oracle Database Testing Guide

Performing Preproduction Validation Steps

Pre-production validation and testing of hardware, software, database, application or any changes is an important way to maintain stability. The high-level pre-production validation steps are:

1. Review the patch or upgrade documentation or any document relevant to that change. Evaluate the possibility of performing a rolling upgrade if your SLAs require zero or minimal downtime. Evaluate any rolling upgrade opportunities to minimize or eliminate planned downtime. Evaluate whether the patch or the change qualifies for Standby-First Patching.

   Note:

Standby-First Patch enables you to apply a patch initially to a physical standby database while the primary database remains at the previous software release (this applies to certain types of software updates and does not apply to major release upgrades; use the Data Guard transient logical standby and DBMS_ROLLING method for patch sets and major releases). Once you are satisfied with the change, then perform a switchover to the standby database. The fallback is to switchback if required. Alternatively, you can proceed to the following step and apply the change to your production environment. For more information, see "Oracle Patch Assurance - Data Guard Standby-First Patch Apply" in My Oracle Support Note 1265700.1 at https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1265700.1

2. Validate the application in a test environment and ensure the change meets or exceeds your functionality, performance, and availability requirements. Automate the procedure and be sure to also document and test a fallback procedure. This requires comparing metrics captured before and after patch application on the test and against metrics captured on the production system. Real Application Testing may be used to capture the workload on the production system and replay it on the test system. AWR and SQL Performance Analyzer may be used to assess performance improvement or regression resulting from the patch.

   Validate the new software on a test system that mimics your production environment, and ensure the change meets or exceeds your functionality, performance, and availability requirements. Automate the patch or upgrade procedure and ensure fallback. Being thorough during this step eliminates most critical issues during and after the patch or upgrade.

3. Use Oracle Real Application Testing and test data management features to comprehensively validate your application while also complying with any security restrictions your line of business may have. Oracle Real Application Testing (a separate database option) enables you to perform real-world testing of Oracle Database. By capturing production workloads and assessing the impact of system changes on these workloads before production deployment, Oracle Real Application Testing minimizes the risk of instabilities associated with system changes. SQL Performance Analyzer and Database Replay are key components of Oracle Real Application Testing. Depending on the nature and impact of the system change being tested, and on the type of system on which the test will be performed, you can use either or both components to perform your testing.

   When performing real-world testing there is a risk of exposing sensitive data to non-production users in a test environment. The test data management features of Oracle Database help to minimize this risk by enabling you to perform data masking and data subsetting on the test data.

4. If applicable, perform final pre-production validation of all changes on a Data Guard standby database before applying them to production. Apply the change in a Data Guard environment, if applicable.

5. Apply the change in your production environment.

See Also:

Data Guard Redo Apply and Standby-First Patching and Data Guard Transient Logical Rolling Upgrades

Converting a Physical Standby Database into a Snapshot Standby Database and Performing a Rolling Upgrade With an Existing Physical Standby Database in Oracle Data Guard Concepts and Administration

Oracle Database Rolling Upgrades: Using a Data Guard Physical Standby Database on http://www.oracle.com/goto/maa

Oracle Patch Assurance - Data Guard Standby-First Patch Apply (Doc ID 1265700.1)

Set Up and Use Security Best Practices

Corporate data can be at grave risk if placed on a system or database that does not have proper security measures in place. A well-defined security policy can help protect your systems from unwanted access and protect sensitive corporate information from sabotage. Proper data protection reduces the chance of outages due to security breaches.

See Also:

Oracle Database Security Guide.

Establish Change Control Procedures

Institute procedures that manage and control changes as a way to maintain the stability of the system and to ensure that no changes are incorporated in the primary database unless they have been rigorously evaluated on your test systems, or any one of the base architectures in the MAA service-level tiers.

Review the changes and get feedback and approval from your change management team.

Apply Recommended Patches and Software Periodically

By periodically testing and applying the latest recommended patches and software versions, you ensure that your system has the latest security and software fixes required to maintain stability and avoid many known issues. Remember to validate all updates and changes on a test system before performing the upgrade on the production system.

Furthermore, Oracle health check tools such as

```
orachk
```

(supporting Non-Engineered Systems and Oracle Database Appliance) and

```
exachk
```

(supporting Engineered Systems such as Oracle Exadata Database Machine, Exalogic, Zero Data Loss Recovery Appliance, and Big Data Appliance) provide Oracle software upgrade advice, critical software update recommendations, and patching and upgrading pre-checks, along with its system and database health checks and MAA recommendations.

> **See Also:**
>
> "Oracle Recommended Patches -- Oracle Database" in My Oracle Support Note 756671.1 at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=756671.1`
>
> "Exadata Database Machine and Exadata Storage Server Supported Versions" in My Oracle Support Note 888828.1 at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=888828.1`
>
> "ORAchk - Health Checks for the Oracle Stack" in My Oracle Support Note 1268927.2 at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1268927.2`
>
> "Oracle Exadata Database Machine exachk or HealthCheck" in My Oracle Support Note 1070954.1 at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1070954.1`

Execute Disaster Recovery Validation

Disaster recovery validation is required to ensure that you meet your disaster recovery service level requirements such as RTO and RPO.

Whether you have a standby database, Oracle GoldenGate replica, or leverage database backups from Zero Data Loss Recovery Appliance (Recovery Appliance), ZFS Storage, or another third party, it is important to ensure that the operations and database administration teams are well prepared to failover or restore the database and application any time the primary database is down or underperforming. The concerned teams should be able to detect and decide to failover or restore as required. Such efficient execution during disasters will significantly reduce overall downtime.

If you use Data Guard or Oracle GoldenGate for high availability, disaster recovery, and data protection, Oracle recommends that you perform regular application and database switchover operations **every three to six months,** or conduct full application and database failover tests.

Periodic RMAN cross checks, RMAN backup validations, and complete database restore and recovery are required to validate your disaster recovery solution through backups. Inherent backup checks and validations are done automatically with the Recovery Appliance, but periodic restore and recovery tests are still recommended.

Establish Escalation Management Procedures

Establish escalation management procedures so repair is not hindered. Most repair solutions, when conducted properly are automatic and transparent with the MAA solution. The challenges occur when the primary database or system is not meeting availability or performance SLAs and failover procedures are not automatic as in the case with some Data Guard failover scenarios. Downtime can be prolonged if proper escalation policies are not followed and decisions are not made quickly.

If availability is the top priority, perform repair and failover operations first and then proceed with gathering logs and information for Root Cause Analysis (RCA) after the application service has been reestablished. For simple data gathering, use the Trace File Analyzer Collector (TFA).

See Also:

MAA web page at `http://www.oracle.com/goto/maa`

My Oracle Support note 1513912.2 “TFA Collector - Tool for Enhanced Diagnostic Gathering” at 1513912.2

Configure Monitoring and Service Request Infrastructure for High Availability

To maintain your High Availability environment, you should configure the monitoring infrastructure that can detect and react to performance and high availability related thresholds before any downtime has occurred.

Also, where available, Oracle can detect failures, dispatch field engineers, and replace failed hardware components such as disks, flash cards, fans, or power supplies without customer involvement.

Run Database Health Checks Periodically

Oracle Database health checks are designed to evaluate your hardware and software configuration and MAA compliance to best practices.

All of the Oracle health check tools will evaluate Oracle Grid Infrastructure, Oracle Database, and provide an automated MAA scorecard or review that highlights when key architectural and configuration settings are not enabled for tolerance of failures or fast recovery. For Oracle's engineered systems such as Exadata Database Machine, there may be hundreds of additional software, fault and configuration checks.

Oracle recommends periodically (for example, monthly for Exadata Database Machine) downloading the latest database health check, executing the health check, and addressing the key FAILURES, WARNINGS, and INFO messages. Use

```
exachk
```

for Engineered Systems such as Oracle Exadata Database Machine, Exalogic, Zero Data Loss Recovery Appliance, and Big Data Appliance, and use

```
orachk
```

for Non-Engineered Systems and Oracle Database Appliance.

Furthermore, it is recommended that you run the health check prior to and after any planned maintenance activity.

You must **evaluate:**

- Existing or new critical health check alerts prior to planned maintenance window
- Adding any new recommendations to the planned maintenance window after testing
- Existing software or critical software recommendations

> **See Also:**
>
> My Oracle Support Note 1268927.2 "ORAchk - Health Checks for the Oracle Stack" at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1268927.2`
>
> My Oracle Support Note 1070954.1 "Oracle Exadata Database Machine exachk or HealthCheck" at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1070954.1`

Configure Oracle Enterprise Manager Monitoring Infrastructure for High Availability

You should configure and use Enterprise Manager and the monitoring infrastructure that detects and reacts to performance and high availability related thresholds to avoid potential downtime.

The monitoring infrastructure assists you with monitoring for High Availability and enables you to do the following:

- Monitor system, network, application, database and storage statistics
- Monitor performance and service statistics
- Create performance and high availability thresholds as early warning indicators of system or application problems
- Provide performance and availability advice
- Established alerts and tools and database performance
- Receive alerts for engineered systems hardware faults

See Also:

MAA Best Practices for Enterprise Manager at http://www.oracle.com/goto/maa

Configure Automatic Service Request Infrastructure

In addition to monitoring infrastructure with Enterprise Manager in the Oracle high availability environment where available, Oracle can detect failures, dispatch field engineers, and replace failing hardware without customer involvement.

For example, Oracle Automatic Service Request (ASR) is a secure, scalable, customer-installable software solution available as a feature. The software resolves problems faster by using auto-case generation for Oracle's Solaris server and storage systems when specific hardware faults occur.

See Also:

See "Oracle Automatic Service Request" in My Oracle Support Note 1185493.1 at `https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1185493.1`

Check the Latest MAA Best Practices

The MAA solution encompasses the full stack of Oracle technologies, so you can find MAA best practices for Oracle Fusion Middleware, Oracle Fusion Applications, Oracle Applications Unlimited, Oracle Exalytics, Oracle Exalogic, Oracle VM, and Oracle Enterprise Manager Cloud Control on the MAA pages.

MAA solutions and best practices continue to be developed and published on http://www.oracle.com/goto/maa.

Part II

Oracle Database High Availability Best Practices

8 Overview of Oracle Database High Availability Best Practices

By adopting the Oracle MAA best practices for Oracle Database, you can achieve the service levels of the Oracle MAA Bronze reference architecture.

The Bronze architecture achieves the highest availability for a single-instance database configuration, whether it is a standalone database or part of a consolidated multitenant database, by using the high availability capabilities included in Oracle Database Enterprise Edition.

The Bronze architecture is the base configuration for the other MAA reference architectures. The Oracle Database best practices should also be implemented in the Silver, Gold, and Platinum references architectures, unless specifically noted in the best practices for that architecture.

For information about the components, service levels, and benefits of the Bronze reference architecture, as well as the MAA architectures that build on the Bronze base, see the "High Availability Reference Architectures" interactive diagram at https://www.oracle.com/webfolder/technetwork/tutorials/architecture-diagrams/high-availability-overview/high-availability-reference-architectures.html.

9 Oracle Database Configuration Best Practices

Adopt the Oracle MAA best practices for configuring all Oracle single-instance databases to reduce or avoid outages, reduce the risk of corruption, and improve recovery performance.

Note that the following Oracle Database best practices are used to configure the Oracle MAA Bronze reference architecture, and they are also the base database base practices for the other MAA reference architectures: Silver (Oracle RAC), Gold (Oracle Data Guard), and Platinum (Oracle GoldenGate).

Use a Server Parameter File (SPFILE)

The server parameter file (SPFILE) enables a single, central parameter file to hold all database initialization parameters associated with all instances of a database. This provides a simple, persistent, and robust environment for managing database parameters. SPFILE is recommended to be placed in the DATA ASM disk group.

Enable Archive Log Mode and Forced Logging

Running the database in `ARCHIVELOG` mode and using database `FORCE LOGGING` mode are prerequisites for database recovery operations.

The `ARCHIVELOG` mode enables online database backup and is necessary to recover the database to a point in time later than what has been restored. Features such as Oracle Data Guard and Flashback Database require that the production database run in `ARCHIVELOG` mode.

If you can isolate data that never needs to be recovered within specific tablespaces, then you can use tablespace level `FORCE LOGGING` attributes instead of the database `FORCE LOGGING` mode.

Configure an Alternate Local Archiving Destination

The local archive destination, usually `LOG_ARCHIVE_DEST_1`, should have an alternate local destination on a different ASM disk group. This configuration prevents database hanging due to lack of archive log space if `DB_RECOVERY_FILE_DEST` fills up or is unavailable for any reason.

Table 9-1 Alternate Local Archiving Configuration Parameters

Database Parameter	LOG_ARCHIVE_DEST_n parameter settings for local archive destinations

LOG_ARCHIVE_DEST_*n*	LOCATION=USE_DB_FILE_RECOVERY_DEST
	VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
	MAX_FAILURE=1
	REOPEN=5
	DB_UNIQUE_NAME=db_unique_name of the database
	ALTERNATE=some other log archive destination. Must be log_archive_dest_[1-10]

LOG_ARCHIVE_DEST_*y*	LOCATION=A disk group other than the disk group used for DB_RECOVERY_FILE_DEST. Usually the DATA disk group.
	VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
	MAX_FAILURE=1
	REOPEN=5
	ALTERNATE= the primary local archive log destination: usually LOG_ARCHIVE_DEST_1

DB_RECOVERY_FILE_DEST	Archive destination, for example, a RECO disk group
LOG_ARCHIVE_DEST_STATE_*n*	ENABLE
LOG_ARCHIVE_DEST_STATE_*y*	ALTERNATE

Sample parameter settings:

- LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_FILE_RECOVERY_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) MAX_FAILURE=1 REOPEN=5 DB_UNIQUE_NAME=*db_unique_name of the database* ALTERNATE=LOG_ARCHIVE_DEST_10'
- LOG_ARCHIVE_DEST_10='LOCATION=+DATA VALID_FOR=(ALL_LOGFILES,ALL_ROLES) MAX_FAILURE=1 REOPEN=5 DB_UNIQUE_NAME=*db_unique_name of the database* ALTERNATE=LOG_ARCHIVE_DEST_1'
- LOG_ARCHIVE_DEST_STATE_1 =enable
- LOG_ARCHIVE_DEST_STATE_10=alternate
- DB_RECOVERY_FILE_DEST=*typically the RECO disk group*

Use a Fast Recovery Area

The Fast Recovery Area is Oracle-managed disk space that provides a centralized disk location for backup and recovery files.

The Fast Recovery Area is defined by setting the following database initialization parameters:

- DB_RECOVERY_FILE_DEST specifies the default location for the fast recovery area. Set this parameter to the RECO disk group.

- DB_RECOVERY_FILE_DEST_SIZE specifies (in bytes) the hard limit on the total space to be used by database recovery files created in the recovery area location.

  Set this parameter to a value large enough to store archived logs, flashback logs and any local database backup files locally. Having the files locally can reduce your recovery time after restoring a backup. RMAN will automatically manage these files according to your RMAN backup and data retention policies. Typically customers store 24 hours of data in the destination

  When your system hosts many databases sharing the same DB_RECOVERY_FILE_DEST_SIZE, space needs to manage and monitored holistically. Recommended to alert when RECO disk group for example is 90% full.

Enable Flashback Database

Flashback Database provides an efficient alternative to point-in-time recovery for reversing unwanted database changes.

Flashback Database lets you rewind an entire database backward in time, reversing the effects of database changes within a time window. The effects are similar to database point-in-time recovery. You can flash back a database by running a single RMAN command or a SQL*Plus statement instead of using a complex procedure.

To enable Flashback Database, configure a fast recovery area and set a flashback retention target using the best practices listed below. This retention target specifies how far back you can rewind a database with Flashback Database.

- Know your application performance baseline before you enable flashback database to help determine the overhead and to assess the application workload implications of enabling flashback database.
- Ensure that the fast recovery area space is sufficient to hold the flashback database flashback logs. A general rule of thumb is that the volume of flashback log generation is approximately the same order of magnitude as redo log generation. For example, if you intend to set DB_FLASHBACK_RETENTION_TARGET to 24 hours, and if the database generates 20 GB of redo in a day, then allow 20 GB to 30 GB disk space for the flashback logs.
  - An additional method to determine fast recovery area sizing is to enable flashback database and allow the database to run for a short period of time (2-3 hours). Query V$FLASHBACK_DATABASE_STAT.ESTIMATED_FLASHBACK_SIZE to retrieve the estimated amount of space required for the fast recovery area.
  - Note that the DB_FLASHBACK_RETENTION_TARGET is a target and there is no guarantee that you can flashback the database that far. In some cases if there is space pressure in the fast recovery area where the flashback logs are stored, then the oldest flashback logs may be deleted. To guarantee a flashback point-in-time you must use guaranteed restore points.
- Ensure that there is sufficient I/O bandwidth to the fast recovery area. Insufficient I/O bandwidth with flashback database on is usually indicated by a high occurrence of the FLASHBACK BUF FREE BY RVWR wait event.
- To monitor the progress of a flashback database operation you can query the V$SESSION_LONGOPS view. An example query to monitor progress is

  ```
  SELECT sofar, totalwork, units FROM v$session_longops WHERE opname =
  'Flashback Database';
  ```

- For repetitive tests where you must flashback to the same point, use flashback database guaranteed restore points instead of enabling flashback database. This will minimize space usage.
- Flashback PDB can rewind a pluggable database without affecting other PDBs in the CDB. You can also create PDB restore points.

Set FAST_START_MTTR_TARGET Initialization Parameter

With Fast-Start Fault Recovery, the `FAST_START_MTTR_TARGET` initialization parameter simplifies the configuration of recovery time from instance or system failure.

The `FAST_START_MTTR_TARGET` parameter specifies a target for the expected recovery time objective (RTO), which is the time, in seconds, that it should take to start the instance and perform cache recovery. When you set this parameter, the database manages incremental checkpoint writes in an attempt to meet the target. If you have chosen a practical value for this parameter, then you can expect your database to recover, on average, in approximately the number of seconds you have chosen.

Initially, set the `FAST_START_MTTR_TARGET` initialization parameter to 300 (seconds), or to the value required for your expected recovery time objective (RTO). As you set or lower this value, database writer (DBWR) will become more active to meet your recovery targets.

Make sure that you have sufficient IO bandwidth to handle potential higher load. See the Database Performance Tuning Guide for information about monitoring and tuning `FAST_START_MTTR_TARGET`.

Outage testing for cases such as node or instance failures during peak loads is recommended.

Protect Against Data Corruption

Oracle Database corruption prevention, detection, and repair capabilities are built on internal knowledge of the data and transactions it protects, and on the intelligent integration of its comprehensive high availability solutions.

A data block is corrupted when it is not in a recognized Oracle Database format, or its contents are not internally consistent. Data block corruption can damage internal Oracle control information or application and user data, leading to crippling loss of critical data and services.

When Oracle Database detects corruption, it offers block media recovery and data file media recovery to recover the data. You can undo database-wide logical corruptions caused by human or application errors with Oracle Flashback Technologies. Tools are also available for proactive validation of logical data structures. For example, the SQL*Plus `ANALYZE TABLE` statement detects inter-block corruptions.

The following are best practices for protecting your database against corruption.

- Use Oracle Automatic Storage Management (Oracle ASM) to provide disk mirroring to protect against disk failures.
- Use the `HIGH` redundancy disk type for optimal corruption repair with Oracle ASM.

  Using Oracle ASM redundancy for disk groups provides mirrored extents that can be used by the database if an I/O error or corruption is encountered. For continued protection, Oracle ASM redundancy lets you move an extent to a different area on a disk if an I/O error occurs. The Oracle ASM redundancy mechanism is useful if you have bad sectors returning media errors.

- Enable Flashback technologies for fast point-in-time recovery from logical corruptions that are most often caused by human error, and for fast reinstatement of a primary database following failover.
- Implement a backup and recovery strategy with Recovery Manager (RMAN) and periodically use the RMAN `BACKUP VALIDATE CHECK LOGICAL` scan to detect corruptions.

  Use RMAN and Oracle Secure Backup for additional block checks during backup and restore operations. Use Zero Data Loss Recovery Appliance for backup and recovery validation including corruption checks and repairs, central backup validation, reduced production database impact, and Enterprise Cloud backup and recovery solutions.
- Set database initialization parameter `DB_BLOCK_CHECKSUM=MEDIUM` or `FULL`.
- Evaluate setting `DB_BLOCK_CHECKING=MEDIUM` or `FULL`, but only after a full performance evaluation with the application.

Set the LOG_BUFFER Initialization Parameter to 128MB or Higher

Set the `LOG_BUFFER` initialization parameter to a minimum of 128 MB for databases with flashback enabled.

Use Automatic Shared Memory Management and Avoid Memory Paging

Enable Automatic Shared Memory Management by setting the `SGA_TARGET` parameter, and set the `USE_LARGE_PAGES` database initialization parameter to `AUTO_ONLY` or `ONLY` and the `USE_LARGE_PAGES` ASM initialization parameter to `TRUE`.

Use the following guidelines in addition to setting `SGA_TARGET` to enable Automatic Shared Memory Management.

- The sum of SGA and PGA memory allocations on the database server should always be less than your system's physical memory while still accommodating memory required for processes, PGA, and other applications running on the same database server.
- To get an accurate understanding of memory use, monitor PGA memory and host-based memory use by querying `V$PGASTAT` for operating systems statistics.
- Avoid memory paging by adjusting the number of databases and applications, or reducing the allocated memory settings.

  Set `PGA_AGGREGATE_LIMIT` to specify a hard limit on PGA memory usage. If the `PGA_AGGREGATE_LIMIT` value is exceeded, Oracle Database first terminates session calls that are consuming the most untunable PGA memory. Then, if the total PGA memory usage is still over the limit, the sessions that are using the most untunable memory will be terminated.

Set the database initialization parameter `USE_LARGE_PAGES=AUTO_ONLY` or `ONLY`, and set the ASM initialization parameter `USE_LARGE_PAGES=TRUE`.

- Make sure that the entire SGA of a database instance is stored in HugePages by setting the `init.ora` parameter `USE_LARGE_PAGES=ONLY`, or set to `AUTO_ONLY` on Exadata systems.

  Setting `USE_LARGE_PAGES=ONLY` is recommended for database instances, because this parameter ensures that an instance will only start when it can get all of its memory for SGA from HugePages.
- For ASM instances leave the parameter `USE_LARGE_PAGES=ONLY` (the default value). This setting still ensures that HugePages are used when available, but also ensures that ASM as part of Grid Infrastructure starts when HugePages are not configured, or insufficiently configured.
- Use Automatic Shared Memory Management, because HugePages are not compatible with Automatic Memory Management.

Use Oracle Clusterware

Oracle Clusterware lets servers communicate with each other, so that they appear to function as a collective unit. Oracle Clusterware has high availability options for all Oracle databases including for single instance Oracle databases. Oracle Clusterware is one of minimum requirements in making applications highly available.

Oracle Clusterware provides the infrastructure necessary to run Oracle Real Application Clusters (Oracle RAC), Oracle RAC One Node, and Oracle Restart. Oracle Grid Infrastructure is the software that provides the infrastructure for an enterprise grid architecture. In a cluster, this software includes Oracle Clusterware and Oracle ASM.

For a standalone server, the Grid Infrastructure includes Oracle Restart and Oracle ASM. Oracle Restart provides managed startup and restart of a single-instance (non-clustered) Oracle database, Oracle ASM instance, service, listener, and any other process running on the server. If an interruption of a service occurs after a hardware or software failure, Oracle Restart automatically restarts the component.

Oracle Clusterware manages resources and resource groups to increase their availability, based on how you configure them. You can configure your resources and resource groups so that Oracle Clusterware:

- Starts resources and resource groups during cluster or server start
- Restarts resources and resource groups when failures occur
- Relocates resources and resource groups to other servers, if the servers are available

For more information, see *Oracle Clusterware Administration and Deployment Guide* topics, High Availability Options for Oracle Database and Making Applications Highly Available Using Oracle Clusterware.

Part III

Oracle Data Guard Best Practices

10 Overview of MAA Best Practices for Oracle Data Guard

By adding a physical standby database with Oracle Active Data Guard, a Silver MAA reference architecture is elevated to a Gold MAA reference architecture. Implement Oracle Data Guard best practices to achieve minimal downtime and potentially zero data loss for all unplanned outages.

Oracle Active Data Guard plays an important role in delivering the high availability and comprehensive data protection that you expect of the Gold MAA reference architecture. The Gold reference architecture, consisting of an Oracle RAC primary database and Oracle RAC standby systems with Oracle Active Data Guard, plus MAA configuration and life cycle operations, provides a comprehensive set of services that create, maintain, manage, and monitor one or more standby databases. Oracle Active Data Guard protects your data during all types of planned maintenance activities, such as software updates and major database upgrades, and unplanned outages, including database failures, site outages, natural disasters, and data corruptions.

The goal of Oracle Data Guard best practices is to help you implement tested and proven MAA best practices to ensure a successful and stable Data Guard deployment. The following steps connect you to the Oracle MAA best practices for planning, implementing, and maintaining this type of architecture.

1. Plan your Oracle Data Guard architecture, and take into account various considerations for the application, network, and so on.
2. Configure and Deploy Data Guard using Oracle MAA best practices.
3. Tune and Troubleshoot your Data Guard deployment.

To learn more about the Gold MAA reference architecture, see High Availability Reference Architectures.

11

Plan an Oracle Data Guard Deployment

Analyze your specific requirements, including both the technical and operational aspects of your IT systems and business processes, understand the availability impact for the Oracle Data Guard architecture options, and consider the impact of your application and network.

Oracle Data Guard Architectures

The Gold MAA reference architecture provides you with four architecture patterns, using Oracle Active Data Guard to eliminate single point of failure. The patterns vary from a single remote active standby with Fast Start Failover and HA Obeserver, to including far sync instances, multiple standbys, and reader farms.

When planning your Gold MAA Reference Architecture, see High Availability Reference Architectures for an overview of each Gold architecture pattern, and choose the elements to incorporate based on your requirements.

Application Considerations for Oracle Data Guard Deployments

As part of planning your Oracle Data Guard deployment, consider the resources required and application availability requirements in a fail over scenario.

Deciding Between Full Site Failover or Seamless Connection Failover

The first step is to evaluate which failover option best meets your business and application requirements when your primary database or primary site is inaccessible or lost due to a disaster.

The following table describes various conditions for each outage type and recommends a failover option in each scenario.

Table 11-1 Recommended Failover Options for Different Outage Scenarios

Outage Type	Condition	Recommended Failover Option
Primary Site Failure (including all application servers)	Primary site contains all existing application servers (or mid-tier servers) that were connected to the failed primary database.	Full site failover is required

Table 11-1 (Cont.) Recommended Failover Options for Different Outage Scenarios

Outage Type	Condition	Recommended Failover Option
Primary Site Failure (with some application servers surviving)	Some or all application servers are not impacted and the surviving application servers can reconnect to new primary database in a secondary disaster recovery site. Application performance and throughput is still acceptable with different network latency between application servers and new primary database in a secondary disaster recovery site. Typically analytical or reporting applications can tolerate higher network latency between client and database without any noticeable performance impact, while OLTP applications performance may suffer more significantly if there is an increase in network latency between the application server and database.	Seamless connection failover is recommended to minimize downtime and enable automatic application and database failover.
Complete Primary Database or Primary Server Failure	Application servers are not impacted and users can reconnect to new primary database in a secondary disaster recovery site. Application performance and throughput is still acceptable with different network latency between application servers and new primary database in a secondary disaster recovery site. Typically analytical or reporting applications can tolerate higher network latency between client and database without any noticeable performance impact, while OLTP applications performance may suffer more significantly if there is an increase in network latency between the application server and database.	If performance is acceptable, seamless connection failover is recommended to minimize downtime and enable automatic application and database failover. Otherwise, full site failover is required.

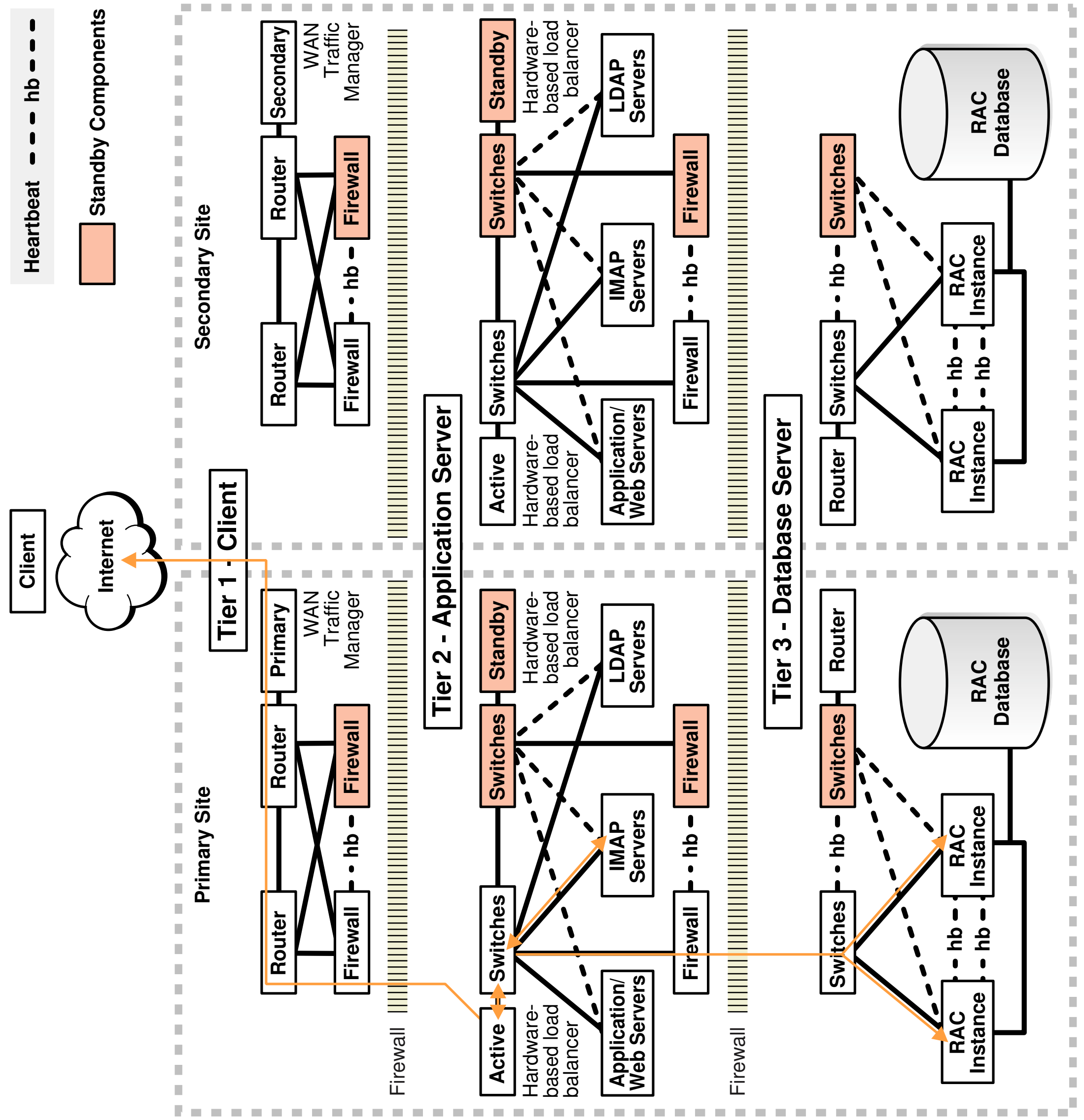
Full Site Failover Best Practices

A **full site failover** means that the complete site fails over to another site with a new set of application tiers and a new primary database.

Complete site failure results in both the application and database tiers becoming unavailable. To maintain availability, application users must be redirected to a secondary site that hosts a redundant application tier and a synchronized copy of the production database.

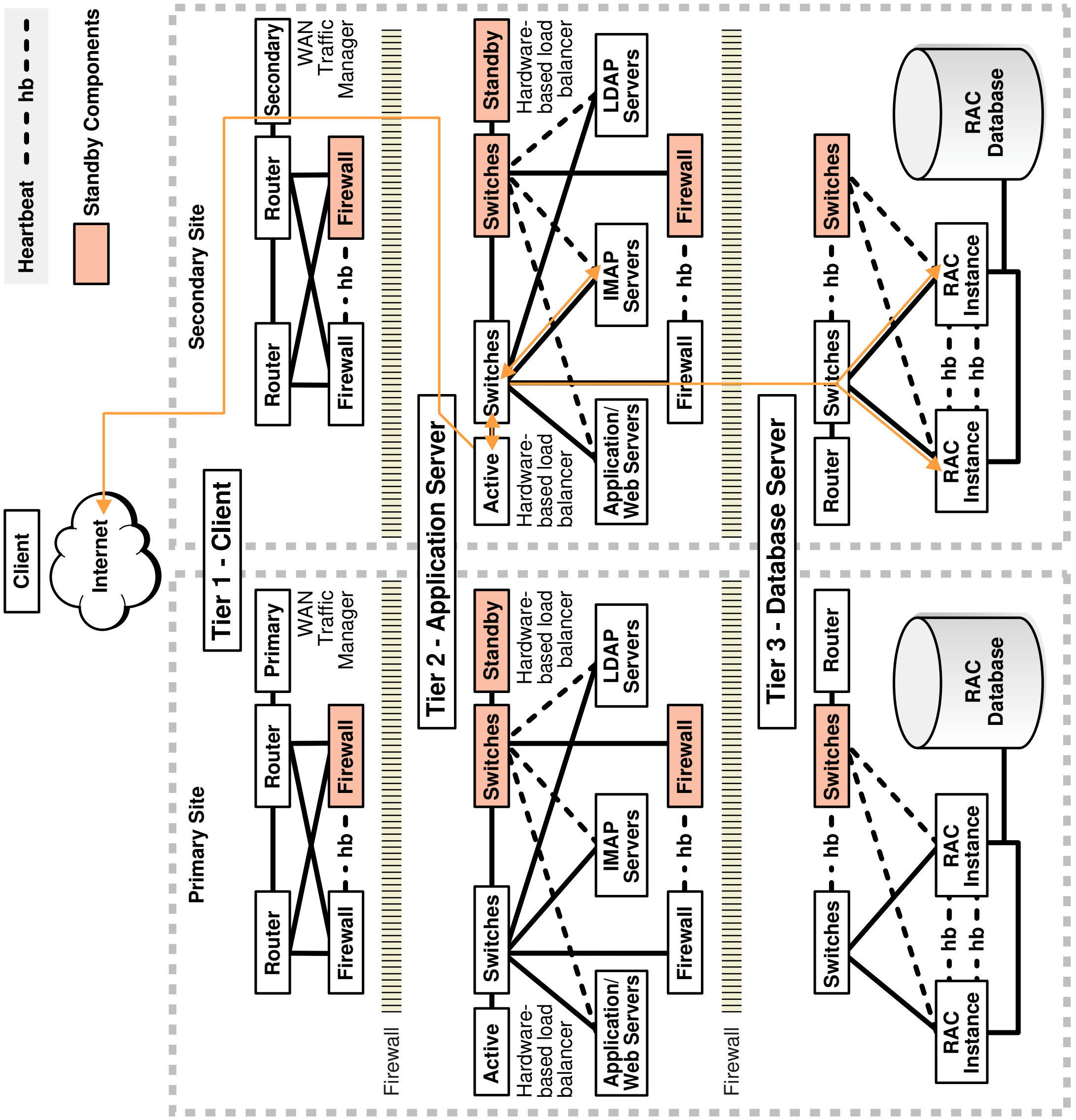
Consider the two figures below. The first figure shows the network routes before failover. Client or application requests enter the Primary site at the client tier, and are routed to the application server and database server tiers on the primary site.

Figure 11-1 Network Routes Before Site Failover



The second figure, below, illustrates the network routes after a complete site failover. Client or application requests enter the Secondary site at the client tier and follow the same path on the secondary site that they followed on the primary site.

Figure 11-2 Network Routes After Site Failover



MAA best practice is to maintain a running application tier at the standby site to avoid incurring start-up time, and to use Oracle Data Guard to maintain a synchronized copy of the production database. Upon site failure, a WAN traffic manager is used to execute a DNS failover (either manually or automatically) to redirect all users to the application tier at standby site while a Data Guard failover transitions the standby database to the primary production role.

Use Oracle Active Data Guard Fast-Start Failover to automate the database failover. Application server and non-database failovers can be automated and coordinated by using Oracle Site Guard. Oracle Site Guard orchestrates and automates any operations, such as starting up application servers on the secondary site, resynchronizing non-database meta data as Data Guard fails over automatically.

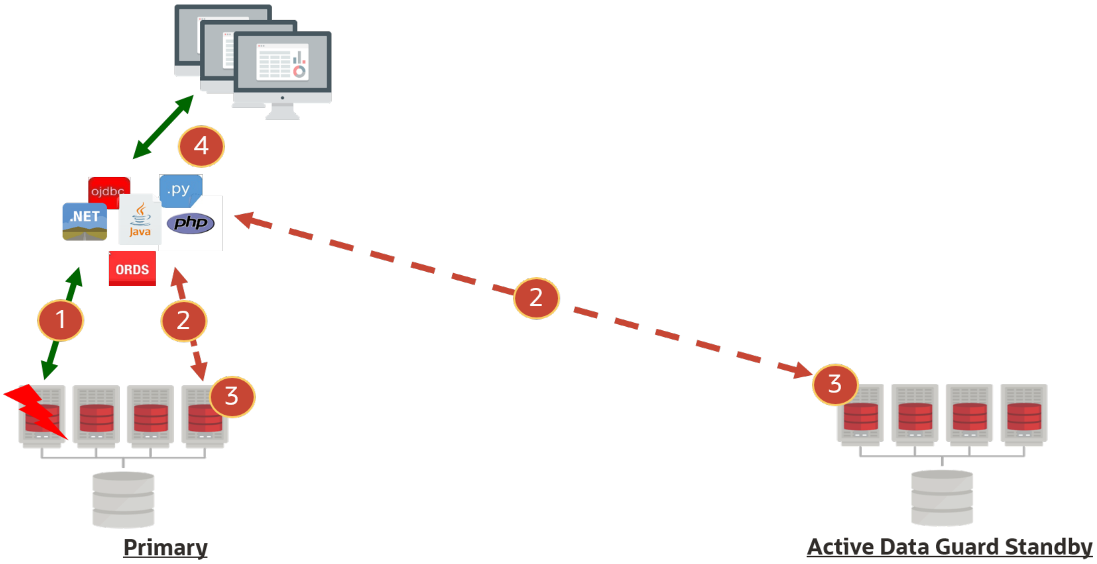
For more information about Oracle Site Guard, see the Oracle Site Guard Administrator's Guide.

Configuring Seamless Connection Failover

Automating seamless client failover in an Oracle Data Guard configuration includes relocating database services to the new primary database as part of a Data Guard failover, notifying clients that a failure has occurred to break them out of TCP timeout, and redirecting clients to the new primary database.

In the following figure, a database request is interrupted by an outage or timeout (1), so the session reconnects to the Oracle RAC cluster (2) (or standby) (2), the database request replays automatically on the alternate node (3), and the result from the database request is returned to the user (4).

Figure 11-3 Seamless Connection Failover



To achieve seamless connection failover, configure Fast Connection Failover (FCF) as a best practice to fully benefit from fast instance and database failover and switchover with Oracle RAC and Oracle Data Guard. FCF enables clients, mid-tier applications, or any program that connects directly to a database to fail over quickly and seamlessly to an available database service when a database service becomes unavailable.

- Use an Oracle Clusterware managed service that is not the default database service (the default service has the same name as the database or PDB). The services that you create provide location transparency and high availability features.
- Use the recommended connection string as discussed in the Application Continuity white paper with built in timeouts, retries, and delays so that incoming connections do not see errors during outages.
- Fast Application Notification (FAN) is a mandatory component to initiate draining, to break out of failures, and to rebalance sessions when services resume and when load imbalances occur. For outages such as node and network failures, fast failover of the application does not happen if the client is not interrupted by FAN. FAN applies to all failover solutions.

- When maintenance starts, drain application connections from the instances or nodes targeted for maintenance. Enable FAN with Universal Connection Pools (UCP) or ODP.NET connection pools or connection tests (or both). Connection pools with FAN are the best solution because pools provide a full life cycle of session movement, that is, the draining and rebalancing of application connection as maintenance progresses.
- The standard solution for failing over sessions is to use Transparent Application Continuity (TAC). Use Transparent Application Failover (TAF) if your application is read-only and does not change Oracle session state in the session after the initial setup. Alternatively, use Application Continuity (AC) if you want to customize how the application connection fails over with side effects or callbacks, or if you have an application that uses state, such as temporary tables, and never cleans up.

When implementing your failover solution, along with the above best practices, follow the instructions in the white paper, Continuous Availability - Application Checklist for Continuous Service for MAA Solutions.

Assessing Network Performance

Oracle Data Guard relies on the underlying network to send redo from the primary and standby databases, and to ensure near zero or zero redo transport lag.

Your network assessment should evaluate the following.

- Network is reliable
- Sufficient available bandwidth to accommodate the maximum redo generation rate and any existing activity sharing the same network
- If using `SYNC` or `FAR SYNC` transport, minimal latency between the primary database and target standby (or far sync server) may be necessary to achieve a tolerable performance impact

Use the `oratcptest` tool to measure and evaluate network bandwidth and latency. The `oratcptest` tool was specifically designed to help you assess network resources to help tune and optimize Data Guard configurations, optimize Data Guard instantiation, help evaluate if `SYNC` transport is viable, and minimize redo transport lags. Using `oratcptest` you can:

- Assess whether you have enough bandwidth available for your Data Guard activity
- Determine optimal TCP socket buffer sizes
- Determine optimal Oracle Net settings
- Tune operating system limits on socket buffer sizes
- Tune Oracle Net session data unit SDU for use with `SYNC`

The following example shows you how `oratcptest` highlights the potential Data Guard asynchronous redo transport bandwidth and the increase in bandwidth by adjusting operating system socket buffer sizes.

First, get the baseline.

Client (primary):

```
$ java -jar oratcptest.jar standby_server -port=port_number -
mode=async -duration=120 -interval=20s -sockbuf=2097152
```

```
[Requesting a test]
        Message payload = 1 Mbyte
        Payload content type = RANDOM
        Delay between messages = NO
        Number of connections = 1
        Socket send buffer = 2 Mbytes
        Transport mode = ASYNC
        Disk write = NO
        Statistics interval = 20 seconds
        Test duration = 2 minutes
        Test frequency = NO
        Network Timeout = NO
        (1 Mbyte = 1024x1024 bytes)
(11:39:16) The server is ready.
                Throughput
(11:39:36) 71.322 Mbytes/s
(11:39:56) 71.376 Mbytes/s
(11:40:16) 72.104 Mbytes/s
(11:40:36) 79.332 Mbytes/s
(11:40:56) 76.426 Mbytes/s
(11:41:16) 68.713 Mbytes/s
(11:41:16) Test finished.
          Socket send buffer = 2097152
             Avg. throughput = 73.209 Mbytes/s
```

Now that you have a baseline with a 2MB buffer, increase the socket buffer size to 4MB to assess any gain in throughput with a larger buffer size.

```
$ java -jar oratcptest.jar -server -port=port_number -sockbuf=4194305
```

Client (primary):

```
$ java -jar oratcptest.jar test.server.address.com -port=<port number> -
mode=async -duration=60s -sockbuf= 4194305
[Requesting a test]
        Message payload = 1 Mbyte
        Payload content type = RANDOM
        Delay between messages = NO
        Number of connections = 1
        Socket send buffer = 4194303 bytes
        Transport mode = ASYNC
        Disk write = NO
        Statistics interval = 10 seconds
        Test duration = 1 minute
        Test frequency = NO
        Network Timeout = NO
        (1 Mbyte = 1024x1024 bytes)

(11:15:06) The server is ready.
                 Throughput
(11:15:16) 113.089 Mbytes/s
(11:15:26) 113.185 Mbytes/s
(11:15:36) 113.169 Mbytes/s
(11:15:46) 113.169 Mbytes/s
```

```
(11:15:56) 113.168 Mbytes/s
(11:16:06) 113.171 Mbytes/s
(11:16:06) Test finished.
          Socket send buffer = 4 Mbytes
             Avg. throughput = 113.149 Mbytes/s
```

The socket buffer size increase yields a 54% improvement in throughput.

See Oracle Support document Assessing and Tuning Network Performance for Data Guard and RMAN (Doc ID 2064368.1) for information about the `oratcptest` tool.

Determining Oracle Data Guard Protection Mode

Oracle Data Guard can run in three different protection modes, which cater to different performance, availability, and data loss requirements. Use this guide to determine which protection mode fits your business requirements and your potential environmental constraints.

Maximum Protection mode guarantees that no data loss will occur if the primary database fails, even in the case of multiple failures (for example, the network between the primary and standby fails, and then at a later time, the primary fails). This policy is enforced by never signaling commit success for a primary database transaction until at least one synchronous Data Guard standby has acknowledged that redo has been hardened to disk. Without such an acknowledgment the primary database will stall and eventually shut down rather than allow unprotected transactions to commit.

To maintain availability in cases where the primary database is operational but the standby database is not, the best practice is to always have a minimum of **two synchronous standby databases** in a Maximum Protection configuration. Primary database availability is not impacted if it receives acknowledgment from at least one synchronous standby database.

Choose this protection mode if zero data loss is more important than database availability. Workload impact analysis is recommended to measure whether any overhead is acceptable when enabling `SYNC` transport.

Maximum Availability mode guarantees that no data loss will occur in cases where the primary database experiences the first failure to impact the configuration. Unlike the Maximum Protection mode, Maximum Availability will wait a maximum of `NET_TIMEOUT` seconds for an acknowledgment from any of the standby databases, after which it will signal commit success to the application and move to the next transaction. Primary database availability (thus the name of the mode) is not impacted by an inability to communicate with the standby (for example, due to standby or network outages). Data Guard will continue to ping the standby and automatically re-establish connection and resynchronize the standby database when possible, but during the period when primary and standby have diverged there will be data loss should a second failure impact the primary database.

For this reason, it is a best practice to monitor protection level, which is simplest using Enterprise Manager Grid Control, and quickly resolve any disruption in communication between the primary and standby before a second failure can occur. This is the most common zero data loss database protection mode.

Choose this protection mode if zero data loss is very important but you want the primary database to continue to be available even with the unlikely case that all standby databases are not reachable. You can complement this solution by integrating multiple standby databases or using Far Sync instances to implement a zero data loss standby solution across a WAN. Workload impact analysis is recommended to measure whether any overhead is acceptable when enabling `SYNC` transport.

Maximum Performance mode is the default Data Guard mode, and it provides the highest level of data protection that is possible without affecting the performance or the availability of the primary database. This is accomplished by allowing a transaction to commit as soon as the redo data needed to recover that transaction is written to the local online redo log at the primary database (the same behavior as if there were no standby database). Data Guard transmits redo concurrently to 1) the standby database directly from the primary log buffer and 2) to the local online redo log write asynchronously enabling a very low potential data loss if the primary site is lost. There is never any wait for standby acknowledgment but the potential data loss for this data protection mode can still be near zero..

Similar to Maximum Availability mode, it is a best practice to monitor the protection level using Enterprise Manager Grid Control, and quickly resolve any disruption in communication between primary and standby before a second failure can occur.

Choose this mode if minimum data loss is acceptable and zero performance impact on the primary is required.

Offloading Queries to a Read-Only Standby Database

Offloading queries and reporting workloads to read-only standby databases can free up your primary database system resources, giving you the ability to add more users, workloads, or even databases.

When you leverage both primary and standby database resources, your business and your applications benefit with higher total system usage, and potentially higher application throughput.

Offload appropriate workloads by following these steps.

1. Identify which application modules are read-only or read-mostly.
   - Evaluate whether you have application services or modules that are read-only.
   - Small and short read-only queries are good candidates to offload to the standby database.
   - Short DMLs, especially those that are response-time sensitive, should not be offloaded to the standby.
   - Large reports or analytic reports are good candidates to offload.
   - Reports that are primarily reads, and that may have an infrequent DML, typically at the start or end of a report, may be good candidates to offload.

     To enable DML Redirection, see ADG_REDIRECT_DML.
2. Gather information about the expected application performance, throughput, response time, or elapsed time service levels for each offload candidate.
   - Once you have determined which queries and reports are good candidates to offload, find out the required expected and maximum response time or elapsed time for each of them. For example some large analytic reports must complete within a 2 hour time span.
   - For short queries, determine the expected response time and throughput expectations.

- These requirements are sometimes referred to as application performance Service Level Agreements, which you need for the next step.

3. Test the performance of each candidate on the standby, and determine whether it meets your requirements.

    - Even though the primary and standby databases have essentially identical data, they are independent databases, independent machines, independent configurations, and have different workloads. For example, an Active Data Guard read-only standby database has a redo apply workload plus the queries that are offloaded, while the primary database may have OLTP, batch, and query workloads.

    - Reported elapsed times, query response time, and workload performance may vary between the primary and standby due to these system, configuration, and workload differences.

    - Tuning requires that you understand system resources, SQL plans, and individual query CPU and wait profile. The tuning recommendations are applicable for both primary and standby databases. See Diagnosing and Tuning Database Performance .

4. Offload a subset of the queries that meet your performance requirements, freeing up resources on the primary database for additional processing capacity.

    - Once you have determined which queries and reports can be offloaded, and the performance of those activities are acceptable, then slowly offload some of the workload and monitor it.

    - Do not oversubscribe and offload too much workload to the standby such that redo apply cannot keep pace after tuning. If the standby falls behind. then you lose that standby as a viable role transition target, and in most cases a standby that lags cannot be used to offload queries.

What if a specific query does not meet your requirements?

1. Consult with a performance engineer and follow the recommendations in Database Performance Tuning Guide.

2. A particular query response time or throughput or report elapsed time is not guaranteed to be the same on the standby system as it was on the primary. Analyze the system resources, SQL plans, overall CPU work time and wait times.

    For example, you may see `standby query scn advance wait` is contributing to a much longer elapsed time in one of your short queries. This wait increase is attributed to Active Data Guard redo apply. If a query sees a certain row in a data block and needs to roll it back because the transaction has not committed as of the query System Commit Number (SCN), it needs to apply corresponding undo to get a consistent read for that query. If the redo for the corresponding undo change has not been applied by redo apply yet, the query needs to wait. The presence of such wait is itself not an issue, and typically may be a couple of milliseconds, but it will vary by workload and may be higher in Real Application Cluster database systems.

12 Configure and Deploy Oracle Data Guard

Use the following Oracle MAA best practice recommendations to configure and deploy Oracle Data Guard.

Oracle Data Guard Configuration Best Practices

The following topics describe Oracle MAA best practices for configuring your Oracle Data Guard configuration.

Apply Oracle Database Configuration Best Practices First

Before you implement the Oracle Data Guard best practices that follow, apply the Oracle Database configuration best practices.

The Oracle Data Guard configuration best practices are considered additional to the general Oracle Database configuration best practices, and will help you achieve the services levels you expect of the MAA Gold reference architecture. It is implied that all of the database configuration best practices should be followed in a Data Guard configuration, and that the Data Guard recommendations discussed here supplant the general database recommendation where there are conflicts.

See Oracle Database Configuration Best Practices for more details.

Use Recovery Manager to Create Standby Databases

There are several methods you can use to create an Oracle Data Guard standby database, but because of its simplicity, the Oracle MAA recommended approach is to create a physical standby database using the `RMAN RESTORE ... FROM SERVICE` clause.

For information about this approach see Creating a Physical Standby database using RMAN restore from service (Doc ID 2283978.1).

Use Oracle Data Guard Broker with Oracle Data Guard

Use Oracle Data Guard broker to create, manage, and monitor an Oracle Data Guard configuration.

You can perform all Data Guard management operations locally or remotely using the broker interfaces: the Data Guard management pages in Oracle Enterprise Manager, which is the broker's graphical user interface (GUI), and the Data Guard command-line interface, called DGMGRL.

The broker interfaces improve usability and centralize management and monitoring of the Data Guard configuration. Available as a feature of Oracle Database Enterprise Edition and Personal Edition, the broker is also integrated with Oracle Database, Oracle Enterprise Manager, and Oracle Cloud Control Plane.

Example Broker Installation and Configuration

The following is an example broker installation and configuration, which is used in all of the broker configuration best practices examples.

Prerequisites:

- Primary database, standby database, and observers reside on separate servers and hardware to provide fault isolation.
- Both primary and standby databases must use an `SPFILE`.
- Set the `DG_BROKER_START` initialization parameter to `TRUE`.
- If any of the databases in the configuration is an Oracle RAC database, you must set up the `DG_BROKER_CONFIG_FILEn` initialization parameters for that database such that they point to the same shared files for all instances of that database. The shared files could be files on a cluster file system, if available, on raw devices, or stored using Oracle Automatic Storage Management.

1. If they do not already exist, create Oracle Net Services aliases that connect to the primary and the standby databases. These aliases should exist in the database home for each host or member of the Data Guard configuration. For Oracle RAC configurations, the aliases should connect using the SCAN name.

```
chicago =
 (DESCRIPTION =
  (ADDRESS_LIST =
    (ADDRESS=(PROTOCOL= TCP)
     (HOST=prmy-scan)(PORT=1521)))
 (CONNECT_DATA =
   (SERVER = DEDICATED)
    (SERVICE_NAME = chicago)))

boston =
 (DESCRIPTION =
  (ADDRESS_LIST =
    (ADDRESS=(PROTOCOL= TCP)
     (HOST=stby-scan)(PORT=1521)))
 (CONNECT_DATA =
   (SERVER = DEDICATED)
    (SERVICE_NAME = boston)))
```

2. On a primary host, connect with DGMGRL and create the configuration.

```
$ dgmgrl sys
Enter password: password
DGMGRL> create configuration 'dg_config' as  primary database is
'chicago' connect identifier is chicago;

 Configuration "dg_config" created with primary database "chicago"

DGMGRL> add database 'boston' as connect identifier is boston;

 Database "boston" added
```

```
DGMGRL> enable configuration;
 Enabled.
```

3. By default the broker sets up a `LOG_ARCHIVE_DEST_`*n* for Maximum Performance database protection mode.

   The broker configures the remote archive destinations with the default values for asynchronous transport, as shown here.

   ```
   log_archive_dest_3=service="boston", ASYNC NOAFFIRM delay=0 optional
   compression=disable max_failure=0 reopen=300 db_unique_name="boston"
   net_timeout=30, valid_for=(online_logfile,all_roles)
   ```

Configure Redo Transport Mode

Configure the redo transport service on each configuration member by setting the `LogXptMode` property to one of the following modes.

- **`ASYNC`** configures redo transport services for this standby database using the `ASYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_`*n* initialization parameter. This mode, along with standby redo log files, enables minimum data loss data protection of potentially less couple seconds with zero performance impact.
- **`FASTSYNC`** configures redo transport services for this standby database using the `SYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_`*n* initialization parameter. Configure synchronous redo transport mode with the `NOAFFIRM` attribute (default=`AFFIRM`) when using maximum availability mode protection mode. This helps to minimize the performance impact of synchronous redo transport by acknowledging the receipt of redo once it has been successfully received and verified within standby memory, but before the redo has been written to the standby redo log. Zero data loss protection is still preserved when only the primary database fails.
- **`SYNC`** configures redo transport services for this standby database using the `SYNC` and `AFFIRM` attributes of the `LOG_ARCHIVE_DEST_`*n* initialization parameter. This mode, along with standby redo log files, is required for configurations operating in either maximum protection mode or maximum availability mode. This redo transport service enables zero data loss data protection to the primary database, but also can incur a higher performance impact if the round trip latency between primary and standby is high (for example, more than 2ms). This option is required for maximum protection mode.

Use the `EDIT DATABASE SET PROPERTY` command to set the transport mode the broker configuration, as shown in these examples.

```
DGMGRL> EDIT DATABASE 'boston' SET PROPERTY LogXptMode=ASYNC;

DGMGRL> EDIT DATABASE 'chicago' SET PROPERTY LogXptMode=FASTSYNC;

DGMGRL> EDIT DATABASE 'SanFran' SET PROPERTY LogXptMode=SYNC;
```

Validate the Broker Configuration

To identify any problems with the overall configuration, validate it using the following steps.

1. Show the status of the broker configuration using the `SHOW CONFIGURATION` command.

   ```
   DGMGRL> show configuration;
   ```

```
 Configuration - dg

   Protection Mode: MaxPerformance
   Members:
   chicago - Primary database
   boston - Physical standby database

 Fast-Start Failover: DISABLED

 Configuration Status:
 SUCCESS   (status updated 18 seconds ago)
```

If the configuration status is SUCCESS, everything in the broker configuration is working properly. However, if the configuration status is WARNING or ERROR then something is wrong in the configuration. Additional error messages that accompany a WARNING or ERROR status can be used to identify the issues. The next step is to examine each database in the configuration to narrow down what the specific error is related to.

2. To identify warnings on the primary and standby databases, show their statuses using the SHOW DATABASE command.

```
DGMGRL> show database chicago

 Database - chicago

   Role:               PRIMARY
   Intended State:     TRANSPORT-ON
   Instance(s):
     tin1
     tin2

 Database Status:
 SUCCESS
```

If the database status is SUCCESS then the database is working properly. However, if database status is WARNING or ERROR, then something is wrong in the database. Additional error messages accompany the WARNING or ERROR status and can be used to identify current issues.

Repeat the SHOW DATABASE command on the standby database and assess any error messages.

3. Validate the databases on Oracle Database 12.1 and later.

In addition to the above commands, in Oracle Database 12.1 and later, the Data Guard broker features a VALIDATE DATABASE command.

```
DGMGRL> validate database chicago

  Database Role:    Primary database
   Ready for Switchover:  Yes

DGMGRL> validate database boston;
```

```
Database Role:     Physical standby database
Primary Database:  tin

Ready for Switchover:  No
Ready for Failover:    Yes (Primary Running)

Capacity Information:
  Database  Instances        Threads
  tin       2                2
  can       1                2
  Warning: the target standby has fewer instances than the
  primary database, this may impact application performance

Standby Apply-Related Information:
  Apply State:      Not Running
  Apply Lag:        Unknown
  Apply Delay:      0 minutes
```

The `VALIDATE DATABASE` command does not provide a `SUCCESS` or `WARNING` status and must be examined to determine if any action needs to be taken.

Configure Fast Start Failover

Fast-start failover allows the broker to automatically fail over to a previously chosen standby database in the event of loss of the primary database. Enabling fast-start failover is requirement to meet stringent RTO requirements in the case of primary database, cluster, or site failure.

Fast-start failover quickly and reliably fails over the target standby database to the primary database role, without requiring you to perform any manual steps to invoke the failover. Fast-start failover can be used only in a broker configuration.

If the primary database has multiple standby databases, then you can specify multiple fast-start failover targets, using the `FastStartFailoverTarget` property. The targets are referred to as candidate targets. The broker selects a target based on the order in which they are specified on the `FastStartFailoverTarget` property. If the designated fast-start failover target develops a problem and cannot be the target of a failover, then the broker automatically changes the fast-start failover target to one of the other candidate targets.

You can use any protection mode with fast-start failover. The maximum protection and maximum availability modes provide an automatic failover environment guaranteed to lose no data. Maximum performance mode provides an automatic failover environment guaranteed to lose no more than the amount of data (in seconds) specified by the `FastStartFailoverLagLimit` configuration property. This property indicates the maximum amount of data loss that is permissible in order for an automatic failover to occur. It is only used when fast-start failover is enabled and the configuration is operating in maximum performance mode.

1. Set the `FastStartFailoverThreshold` property to specify the number of seconds you want the observer and target standby database to wait, after detecting the primary database is unavailable, before initiating a failover, as shown in this example.

   ```
   DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverThreshold =
   seconds;
   ```

A fast-start failover occurs when the observer and the standby database both lose contact with the production database for a period of time that exceeds the value set for `FastStartFailoverThreshold`, and when both parties agree that the state of the configuration is synchronized (Maximum Availability), or that the lag is not more than the configured `FastStartFailoverLagLimit` (Maximum Performance).

An optimum value for `FastStartFailoverThreshold` weighs the trade-off between the fastest possible failover (minimizing downtime) and unnecessarily triggering failover because of temporary network irregularities or other short-lived events that do not have material impact on availability.

The default value for `FastStartFailoverThreshold` is 30 seconds.

The following table shows the recommended settings for `FastStartFailoverThreshold` in different use cases.

Table 12-1 Minimum Recommended Settings for FastStartFailoverThreshold

Configuration	minimum Recommended Setting
Single-instance primary, low latency, and a reliable network	15 seconds
Single-instance primary and a high latency network over WAN	30 seconds
Oracle RAC primary	Oracle RAC miscount + reconfiguration time + 30 seconds

2. Determine where to place the observer in your topology.

   In an ideal state fast-start failover is deployed with the primary, standby, and observer, each within their own availability domain (AD) or data center; however, configurations that only use two availability domains, or even a single availability domain, must be supported. The following are observer placement recommendations for two use cases.

   Deployment Configuration 1: 2 regions with two ADs in each region.

   - Initial primary region has the primary database in AD1, and two high availability observers (one observer in AD2 and second HA observer in AD1)
   - Initial standby region has the standby database in AD1, and two high availability observers used after role change (one observer in AD2 and second HA observer in AD1)
   - For the observer, MAA recommends at least 2 observer targets in the same primary region but in different ADs

   Deployment Configuration 2: 2 regions with only 1 AD in each region

   - Initial primary regions have the primary database and two light weight servers to host observers
   - Initial standby region has the standby database and two light weight servers to host observers (when there is a role change)

3. Configure observer high availability.

   You can register up to three observers to monitor a single Data Guard broker configuration. Each observer is identified by a name that you supply when you issue the START OBSERVER command. You can also start the observers as a background process.

```
DGMGRL> sys@boston
Enter password: password
DGMGRL> start observer number_one in background;
```

On the same host or a different host you can start additional observers for high availability:

```
DGMGRL> sys@boston
Enter password: password
DGMGRL> start observer number_two in background;
```

Only the primary observer can coordinate fast-start failover with Data Guard broker. All other registered observers are considered to be backup observers.

If the observer was not placed in the background then the observer is a continuously executing process that is created when the START OBSERVER command is issued. Therefore, the command-line prompt on the observer computer does not return until you issue the STOP OBSERVER command from another DGMGRL session. To issue commands and interact with the broker configuration, you must connect using another DGMGRL client session.

Now that you have correctly configured fast-start failover, the following conditions can trigger a failover.

- Database failure where all database instances are down
- Data files taken offline because of I/O errors
- Both the Observer and the standby database lose their network connection to the production database, and the standby database confirms that it is in a synchronized state
- A user-configurable condition

Optionally, you can specify the following conditions for which a fast-start failover can be invoked. It is recommend that you leave these user-configurable conditions at the default values and not invoke an automatic failover.

- Data file offline (write error)
- Corrupted Dictionary
- Corrupted Control file
- Inaccessible Log file
- Stuck Archiver
- ORA-240 (control file enqueue timeout)

Should one of these conditions be detected, the observer fails over to the standby, and the primary shuts down, regardless of how FastStartFailoverPmyShutdown is set. Note that the for user-configurable conditions, the fast-start failover threshold is ignored and the failover proceeds immediately.

Fast Start Failover with Multiple Standby Databases

The `FastStartFailoverTarget` configuration property specifies the `DB_UNIQUE_NAME` of one or more standby databases that can act as target databases in a fast-start failover situation when the database on which the property is set is the primary database. These possible target databases are referred to as candidate fast-start failover targets.

The `FastStartFailoverTarget` configuration property can only be set to the name of physical standbys. It cannot be set to the name of a snapshot standby database, far sync instance, or Zero Data Loss Recovery Appliance.

If only one physical standby database exists, then the broker selects that as the default value for this property on the primary database when fast-start failover is enabled. If more than one physical standby database exists, then the broker selects one based on the order in which they are specified in the property definition. Targets are verified when fast-start failover is enabled

Set Send and Receive Buffer Sizes

For optimal network throughput, the minimum recommended settings for TCP send and receive socket buffer sizes is a value equal to the bandwidth-delay product (BDP) of the network link between the primary and standby systems.

Settings higher than the BDP may also yield incremental improvement. For example, Oracle MAA tests simulating high-latency, high-bandwidth networks continued to realize small, incremental increases in throughput as TCP send and receive socket buffer settings were increased to 3xBDP.

BDP is product of the network bandwidth and latency. Socket buffer sizes are set using the Oracle Net Services parameters `RECV_BUF_SIZE` and `SEND_BUF_SIZE`, so that the socket buffer size setting affects only Oracle TCP connections.

The operating system may impose limits on the socket buffer size that must be adjusted so Oracle can use larger values. For example, on Linux, the parameters `net.core.rmem_max` and `net.core.wmem_max` limit the socket buffer size and must be set larger than `RECV_BUF_SIZE` and `SEND_BUF_SIZE`.

For example, if bandwidth is 622 Mbits and latency is 30 ms, then you would calculate the minimum size for the `RECV_BUF_SIZE` and `SEND_BUF_SIZE` parameters as shown here.

Bandwidth Delay Product (BDP) = bandwidth x latency

BDP = 622,000,000 (bandwidth) / 8 x 0.030 (latency) = 2,332,500 bytes.

Given this example the optimal send and receive socket buffer sizes are calculated as follows.

Socket buffer size = 3 x BDP

= 2,332,500 (BDP) x 3

= 6,997,500 bytes

The size of the socket buffers can be set at the operating system level or at the Oracle Net Services level. As socket buffer size requirements can become quite large (depending on network conditions) it is recommended that you set them at the Oracle

Net Services level so that normal TCP sessions, such as telnet, do not use additional memory.

Note that some operating systems have parameters that set the maximum size for all send and receive socket buffers. You must ensure that these values have been adjusted to allow Oracle Net Services to use a larger socket buffer size.

With Oracle Net Services you can set the send and receive socket buffer sizes globally for all connections using the following parameters in the `sqlnet.ora` file.

```
RECV_BUF_SIZE=6997500
```

```
SEND_BUF_SIZE=6997500
```

If you only want the larger buffer sizes for the connections associated with Data Guard transport then configure `RECV_BUF_SIZE` and `SEND_BUF_SIZE` in the Oracle Net Services alias used for transport as well as in the listener on the standby host.

The following example shows the send and receive socket buffer size set as a description attribute for a particular connect descriptor.

```
standby =
 (DESCRIPTION=
   (SEND_BUF_SIZE=6997500)
   (RECV_BUF_SIZE=6997500)
     (ADDRESS=(PROTOCOL=tcp)
     (HOST=stby_host)(PORT=1521))
   (CONNECT_DATA=
     (SERVICE_NAME=standby)))
```

The socket buffer sizes must be configured the same for all databases within a Data Guard configuration. On a standby side or the receiving side you can do this in either the `sqlnet.ora` or the `listener.ora` file.

In the `listener.ora` file, you can either specify the buffer space parameters for a particular protocol address or for a description, as shown here.

```
LISTENER =
  (DESCRIPTION=
    (ADDRESS=(PROTOCOL=tcp)
    (HOST=stby_host)(PORT=1521)
    (SEND_BUF_SIZE=9375000)
    (RECV_BUF_SIZE=9375000)))
```

Set SDU Size to 65535 for Synchronous Transport Only

With Oracle Net Services you can control data transfer by adjusting the session data unit (SDU) size. Oracle testing has shown that setting the SDU parameter to its maximum value of 65535 improves performance of synchronous transport.

You can set SDU on a per connection basis using the SDU parameter in the local naming configuration file, `tnsnames.ora`, and the listener configuration file, `listener.ora`, or you can set the SDU for all Oracle Net Services connections with the profile parameter `DEFAULT_SDU_SIZE` in the `sqlnet.ora` file.

Configure Online Redo Logs Appropriately

Redo log switching has a significant impact on redo transport and apply performance. Follow these best practices for sizing the online redo logs on the primary and standby databases.

Following these guidelines for online redo logs.

- All online redo log groups should have identically sized logs (to the byte).
- Online redo logs should reside on high performing disks (DATA disk groups).
- Create a minimum of three online redo log groups per thread of redo on Oracle RAC instances.
- Create online redo log groups on shared disks in an Oracle RAC environment.
- Multiplex online redo logs (multiple members per log group) unless they are placed on high redundancy disk groups.
- Size online redo logs to switch no more than 12 times per hour (every ~5 minutes). In most cases a log switch every 15 to 20 minutes is optimal even during peak workloads.

Sizing Redo Logs

Size the redo logs based on the peak redo generation rate of the primary database.

You can determine the peak rate by running the query below for a period of time that includes the peak workload. The peak rate could be seen at month-end, quarter-end, or annually. Size the redo logs to handle the highest rate in order for redo apply to perform consistently during these workloads.

```
SQL> SELECT thread#,sequence#,blocks*block_size/1024/1024 MB,
(next_time-first_time)*86400 sec,
 blocks*block_size/1024/1024)/((next_time-first_time)*86400) "MB/s"
 FROM v$archived_log WHERE ((next_time-first_time)*86400<>0) and
first_time
 between to_date('2015/01/15 08:00:00','YYYY/MM/DD HH24:MI:SS')
 and to_date('2015/01/15 11:00:00','YYYY/MM/DD HH24:MI:SS') and
dest_id=1 order by first_time;

   THREAD#  SEQUENCE#         MB        SEC       MB/s
---------- ---------- ---------- ---------- ----------
         2       2291 29366.1963        831  35.338383
         1       2565 29365.6553        781 37.6000708
         2       2292 29359.3403        537  54.672887
         1       2566 29407.8296        813 36.1719921
         2       2293 29389.7012        678 43.3476418
         2       2294 29325.2217       1236 23.7259075
         1       2567 11407.3379       2658 4.29169973
         2       2295 29452.4648        477 61.7452093
         2       2296 29359.4458        954 30.7751004
         2       2297 29311.3638        586 50.0193921
         1       2568 3867.44092       5510 .701894903
```

Choose the redo log size based on the peak generation rate with the following chart.

Table 12-2 Recommended Redo Log Size

Peak Redo Rate	Recommended Redo Log Size
<= 1 MB/s	1 GB
<= 5 MB/s	4 GB
<= 25 MB/s	16 GB
<= 50 MB/s	32 GB
> 50 MB/s	64 GB

Use Standby Redo Log Groups

Configure the standby redo log groups on all primary and standby databases for improved availability and performance.

For each redo log thread--a thread is associated with an Oracle RAC database instance--the number of standby redo log groups must be greater than or equal to (>=) the number of online redo log groups.

Consider the following additional guidelines when creating standby redo log groups.

- All online redo logs and standby redo log groups should have identically sized logs (to the byte). Standby redo logs are not used if they are not the same size as the online redo logs.
- All standby redo log groups should have identically sized logs (to the byte) on both the primary and standby databases.
- Standby redo logs should reside on high performing disks (DATA disk group).
- Standby redo logs should be multiplexed (multiple members per log group) unless placed on high redundancy disk groups. Multiplexing standby redo logs is optional in all cases because Data Guard can fetch any missing redo.
- In an Oracle RAC environment, create standby redo logs on a shared disk.
- In an Oracle RAC environment, assign a thread to each standby redo log group.

The following example creates three log groups for each redo thread.

```
SQL> ALTER DATABASE ADD STANDBY LOGFILE THREAD 1 GROUP 7 ('+DATA') SIZE
4194304000, GROUP 8 ('+DATA') SIZE 4194304000, GROUP 9 ('+DATA') SIZE
4194304000;

SQL> ALTER DATABASE ADD STANDBY LOGFILE THREAD 2 GROUP 10 ('+DATA') SIZE
4194304000, GROUP 11 ('+DATA') SIZE 4194304000, GROUP 12 ('+DATA') SIZE
419430400
```

To check the thread number and group numbers of the online redo logs, query the `V$LOG` view.

```
SQL> SELECT * FROM V$LOG;
```

To check the results of the ALTER DATABASE ADD STANDBY LOGFILE THREAD statements, query the V$STANDBY_LOG view.

```
SQL> SELECT * FROM V$STANDBY_LOG;
```

Protect Against Data Corruption

Oracle Database corruption prevention, detection, and repair capabilities are built on internal knowledge of the data and transactions it protects, and on the intelligent integration of its comprehensive high availability solutions.

When data corruption is detected, Oracle Data Guard, block media recovery, and data file media recovery can recover the data. Database-wide logical corruptions caused by human or application errors can be undone with Oracle Flashback Technologies.

Tools are also available for proactive validation of logical data structures. For example, the SQL*Plus ANALYZE TABLE statement detects inter-block corruptions.

Achieve the most comprehensive data corruption prevention and detection with these best practices.

- Use Oracle Data Guard with physical standby databases to prevent widespread block corruption. Oracle Data Guard is the best solution for protecting Oracle data against data loss and corruption, and lost writes.
- Set the Oracle Database block-corruption initialization parameters on the Data Guard primary and standby databases as shown in the following table.

Table 12-3 Block-Corruption Initialization Parameter Settings

On the primary database set...	On the standby databases set...
DB_BLOCK_CHECKSUM=MEDIUM or FULL	DB_BLOCK_CHECKSUM=MEDIUM or FULL
DB_LOST_WRITE_PROTECT=TYPICAL	DB_LOST_WRITE_PROTECT=TYPICAL
DB_BLOCK_CHECKING=FALSE*	DB_BLOCK_CHECKING=MEDIUM or FULL

* DB_BLOCK_CHECKING on the PRIMARY is recommended to be set to MEDIUM or FULL but only after a full performance evaluation with the application.

- Performance overhead is incurred on every block change, therefore performance testing is of particular importance when setting the DB_BLOCK_CHECKING parameter. Oracle highly recommends the minimum setting of DB_BLOCK_CHECKING=MEDIUM (block checks on data blocks but not index blocks) on either the primary or standby database. If the performance overhead of enabling DB_BLOCK_CHECKING to MEDIUM or FULL is unacceptable on your primary database, then set DB_BLOCK_CHECKING to MEDIUM or FULL for your standby databases.

The following recommendations also help to protect against data corruptions.

- Use Oracle Automatic Storage Management (Oracle ASM) to provide disk mirroring to protect against disk failures.
- Use Oracle ASM HIGH REDUNDANCY for optimal corruption repair. Using Oracle ASM redundancy for disk groups provides mirrored extents that can be used by the database if an I/O error or corruption is encountered. For continued protection, Oracle ASM redundancy provides the ability to move an extent to a different area

on a disk if an I/O error occurs. The Oracle ASM redundancy mechanism is useful if you have bad sectors returning media errors.

- Enable Flashback Technologies for fast point-in-time recovery from logical corruptions most often caused by human error and for fast reinstatement of a primary database following failover.
- Use RMAN for additional block checks during backup and restore operations. Implement a backup and recovery strategy with Recovery Manager (RMAN) and periodically use the RMAN `BACKUP VALIDATE CHECK LOGICAL` scan to detect corruptions.
- Use Zero Data Loss Recovery Appliance for backup and recovery validation including corruption checks and repairs, central backup validation, reduced production database impact, and Enterprise Cloud backup and recovery solution.

Use Flashback Database for Reinstatement After Failover

Enable Flashback Database on both the primary and standby database, so that if the original primary database has not been damaged, you can reinstate the original primary database as a new standby database following a failover.

If there is a failure during the switchover process, then it can easily be reversed when Flashback Database is enabled.

Set `DB_FLASHBACK_RETENTION_TARGET` to the same value on the standby database as the primary. Set `DB_FLASHBACK_RETENTION_TARGET` initialization parameter to the largest value prescribed by any of the following conditions that apply.

- To leverage flashback database to reinstate your failed primary database after Data Guard failover, for most cases set `DB_FLASHBACK_RETENTION_TARGET` to a minimum of 120 (minutes) to enable reinstatement of a failed primary.
- If using Flashback Database for fast point in time recovery from user error or logical corruptions, set `DB_FLASHBACK_RETENTION_TARGET` to a value equal to the farthest time in the past to which the database should be recovered. If you can detect and repair from logical corruptions in less than 24 hours, then set `DB_FLASHBACK_RETENTION_TARGET` to a minimum of 1440 (minutes).

Use Force Logging Mode

When the primary database is in `FORCE LOGGING` mode, all database data changes are logged. `FORCE LOGGING` mode ensures that the standby database remains consistent with the primary database.

If it is not possible to use this mode because you require the load performance with `NOLOGGING` operations, then see Enable an Appropriate Logging Mode for other options.

You can enable force logging immediately by issuing an `ALTER DATABASE FORCE LOGGING` statement. If you specify `FORCE LOGGING`, then Oracle waits for all ongoing non-logged operations to finish.

Configuring Multiple Standby Databases

An Oracle Data Guard configuration with multiple standby databases gives you the benefits of both local and remote standby databases.

A local standby database can provide zero data loss failover and application downtime reduced to seconds. If a regional disaster occurs, making the primary and local standby systems inaccessible, the application and database can fail over to the remote standby. See "Gold: Multiple Standby Databases" for a full discussion of the features and benefits of a multiple standby configuration.

Managing Oracle Data Guard Configurations with Multiple Standby Databases

The Oracle Data Guard broker automates management and operation tasks across multiple databases in an Oracle Data Guard configuration. The broker also monitors all of the systems in a single Oracle Data Guard configuration.

In a multi-member Data Guard configuration the following redo transport destinations are supported:

- Oracle Data Guard standby databases
- Far sync instances (See Using Far Sync Instances for more information)
- Oracle Streams downstream capture databases
- Zero Data Loss Recovery Appliance (Recovery Appliance)

Multiple Standby Databases and Redo Routes

You can use the Oracle Data Guard broker `RedoRoutes` property to override the default behavior by which a primary database sends the redo that it generates to every other redo transport destination in the configuration.

An example redo transport topology that differs from the default would be one in which a physical standby, or a far sync instance, forwards redo received from the primary database to one or more destinations, or one in which the redo transport mode used for a given destination is dependent on which database is in the primary role.

Consider a configuration that has a primary database (North_Sales) and two physical standby databases (Local_Sales and Remote_Sales). The Local_Sales database is located in the same data center as the primary for high availability purposes and for simpler application and database failover. The Remote_Sales database is located in a remote data center for disaster recovery purposes.

Rather than have North_Sales ship its redo to both databases, you can use the `RedoRoutes` broker property to configure real-time cascading, in which the local physical standby database forwards to Remote_Sales the redo it receives from North_Sales. To accomplish this, the `RedoRoutes` property is set on North_Sales and Local_Sales as follows:

- On the North_Sales database, the `RedoRoutes` property specifies that if North_Sales is in the primary role, then it should ship redo to the Local_Sales database using synchronous transport mode. This rule prevents the primary from shipping redo data directly to the Remote_Sales database.
- On the Local_Sales database, the `RedoRoutes` property must specify that if North_Sales is in the primary role, then Local_Sales should forward redo it receives from North_Sales on to Remote_Sales.

To see the runtime `RedoRoutes` configuration, use the `SHOW CONFIGURATION` command. For example:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - Sales_Configuration

  Protection Mode: MaxAvailability
  Members:
 North_Sales  - Primary database
    Local_Sales  - Physical standby database
      Remote_Sales  - Physical standby database (receiving current redo)

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

Note that the asynchronous redo transport attribute was explicitly specified in the redo route rule for the Remote_Sales destination to enable real-time cascading of redo to that destination. (Real-time cascading requires a license for the Oracle Active Data Guard option.)

To disable real-time cascading of redo, do not specify the asynchronous redo transport attribute. For example:

```
DGMGRL> EDIT DATABASE 'Local_Sales' SET PROPERTY 'RedoRoutes' =
'(North_Sales : Remote_Sales)';
```

See RedoRoutes for more information.

Using the RedoRoutes Property for Remote Alternate Destinations

The `RedoRoutes` property can be used to set up a remote alternate destination, so that a terminal member can still receive redo data even if the member from which it was receiving the redo data fails.

Using the previous example, you can have the primary database, North_Sales, send redo data directly to Remote_Sales if the Local_Sales standby database failed. It is also possible, using the `PRIORITY` attribute, to specify that once the Local_Sales failure has been resolved it can resume shipping redo to Remote_Sales.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY
 'RedoRoutes' = '(LOCAL : ( Local_Sales ASYNC PRIORITY=1, Remote_Sales ASYNC
PRIORITY=2 ))';
Property "RedoRoutes" updated

DGMGRL> EDIT DATABASE 'Local_Sales'
 SET PROPERTY 'RedoRoutes' = '(North_Sales : Remote_Sales ASYNC)';
Property "RedoRoutes" updated

DGMGRL> SHOW CONFIGURATION;

Configuration - Sales_Configuration
```

```
  Protection Mode: MaxPerformance
  Members:
  North_Sales    - Primary database
    Local_Sales  - Physical standby database
      Remote_Sales - Physical standby database
Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

To see the full `RedoRoutes` configuration, use the `SHOW CONFIGURATION VERBOSE` command. For example:

```
DGMGRL> SHOW CONFIGURATION VERBOSE;

Configuration - Sales_Configuration

  Protection Mode: MaxPerformance
  Members:
    North_Sales - Primary database
      Local_Sales - Physical standby database
        Remote_Sales - Physical standby database
      Remote_Sales - Physical standby database (alternate of
Local_Sales)

  Properties:
    FastStartFailoverThreshold      = '180'
    OperationTimeout                = '30'
    TraceLevel                      = 'USER'
    FastStartFailoverLagLimit       = '300'
    CommunicationTimeout            = '180'
    ObserverReconnect               = '0'
    FastStartFailoverAutoReinstate  = 'TRUE'
    FastStartFailoverPmyShutdown    = 'TRUE'
    BystandersFollowRoleChange      = 'ALL'
    ObserverOverride                = 'FALSE'
    ExternalDestination1            = ''
    ExternalDestination2            = ''
    PrimaryLostWriteAction          = 'CONTINUE'
    ConfigurationWideServiceName    = 'c0_CFG'

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

Fast Start Failover with Multiple Standby Databases

The Oracle Data Guard `FastStartFailoverTarget` broker configuration property specifies the `DB_UNIQUE_NAME` of one or more standby databases that can act as target databases in a fast-start failover scenario when the database on which the property is set is the primary database.

These possible target databases are referred to as *candidate fast-start failover targets*. The `FastStartFailoverTarget` property can only be set to the name of physical standbys. It cannot be set to the name of a snapshot standby database, far sync instance, or Zero Data Loss Recovery Appliance.

If only one physical standby database exists, then the broker selects that database as the default value for `FastStartFailoverTarget` on the primary database when fast-start failover is enabled. If more than one physical standby database exists, then the broker selects a single standby based on the order in which they are specified in the property definition. The targets are verified when fast-start failover is enabled.

See also, FastStartFailoverTarget.

Setting FastStartFailoverTarget

If you have two or more standby databases, set up the `FastStartFailoverTarget` configuration property on the primary database to indicate the desired fast-start failover target standby database.

The Oracle Data Guard broker reciprocally sets this property for the target standby database to indicate the primary database as its future target standby database when fast-start failover is actually enabled. There is no need for you set this property on the target standby as this is done for you automatically. For example:

```
DGMGRL> edit database moe set property ='curly,larry';
Property "faststartfailovertarget" updated
```

After `FastStartFailoverTarget` is configured, continue with enabling fast-start failover. When fast-start failover is enabled, you cannot change the `FastStartFailoverTarget` configuration property on the primary or target standby databases.

To change the `FastStartFailoverTarget` property to point to a different standby database, disable fast-start failover, set the `FastStartFailoverTarget` property, and reenable fast-start failover. This action does not impact primary or standby database availability or up time.

Switchover with FastStartFailoverTarget Set

If fast-start failover is enabled with `FastStartFailoverTarget` set you can still perform a switchover or a manual failover, as long the role change is directed to the same standby database that was specified for the `FastStartFailoverTarget` database property on the primary database.

Attempting to switch over to a standby that is not the fast-start failover target results in ORA-16655.

```
DGMGRL> switchover to curly
Performing switchover NOW, please wait...
Error: ORA-16655: specified standby database not the current fast-start
failover target standby
```

To switch over to a standby that is not the primary fast-start target:

1. Disable fast-start failover.

   ```
   DGMGRL> DISABLE FAST_START FAILOVER;
   ```

2. Edit the `FastStartFailoverTarget` property to list the standby you wish to switch over to first.

   ```
   DGMGRL> edit database moe set property
   FastStartFailoverTarget='curly,larry';
   Property "faststartfailovertarget" updated
   ```

3. Enable fast-start failover.

   ```
   DGMGRL> ENABLE FAST_START FAILOVER;
   ```

4. Perform the switchover operation.

   ```
   DGMGRL> switchover to curly
   Performing switchover NOW, please wait...
   ```

Fast-Start Failover Outage Handling

If the primary database's fast-start failover target standby database becomes unavailable, perhaps because the standby database or instance is down or there's an issue with transporting redo, then the primary's fast-start failover target is automatically switched to the next target configured in the `FastStartFailoverTarget` property.

Note that is can take several ping cycles to effect the target switch: one ping to recognize that the current target is not viable, and another ping to propose the target switch and finalize it.

If the original fast-start failover target comes back online, a switch back to the original target is not performed automatically. To get the original target back after an outage you must disable and then enable fast-start failover.

Oracle Active Data Guard Far Sync Solution

To support zero data loss, you can deploy between the primary and standby databases an Oracle Data Guard far sync instance, which is a remote Oracle Data Guard destination that accepts redo from the primary database and then ships that redo to other members of the Oracle Data Guard configuration.

About Far Sync

Far Sync is an Oracle Active Data Guard feature that provides increased flexibility in the location of a disaster recovery site for those who wish to implement zero data loss protection.

Even users who have already deployed Oracle Data Guard synchronous transport can benefit from configuring a far sync instance closer to the primary than their current standby to reduce the performance impact on the production database.

Synchronous redo transport over WAN distances or on an under-performing network often has too large an impact on primary database performance to support zero data loss protection. Oracle Active Data Guard Far Sync provides the ability to perform a zero data loss failover to a remote standby database without requiring a second standby database or complex operation.

Far Sync enables this by deploying a far sync instance (a lightweight Oracle instance) at a distance that is within an acceptable range of the primary for synchronous redo transport. A far sync instance receives redo from the primary using synchronous transport and forwards the redo to up to 29 remote standby databases using asynchronous transport.

Far sync instances are part of the Oracle Active Data Guard Far Sync feature, which requires an Oracle Active Data Guard license.

Offloading to a Far Sync Instance

A far sync instance offloads from the primary any overhead of resolving gaps in redo received by the remote standby database (for example, following network or standby database outages) and can conserve WAN bandwidth by performing redo transport compression without impacting primary database performance.

Note that redo compression requires that the Advanced Compression Option be licensed.

Redo Transport Encryption can additionally be offloaded to the far sync instance. Including Advanced Security Option (ASO) encryption during MAA testing showed no impact to the performance of the primary nor currency of the standby databases.

Oracle recommends using ASO for encryption because it is tested and integrated with Oracle Net and Data Guard.

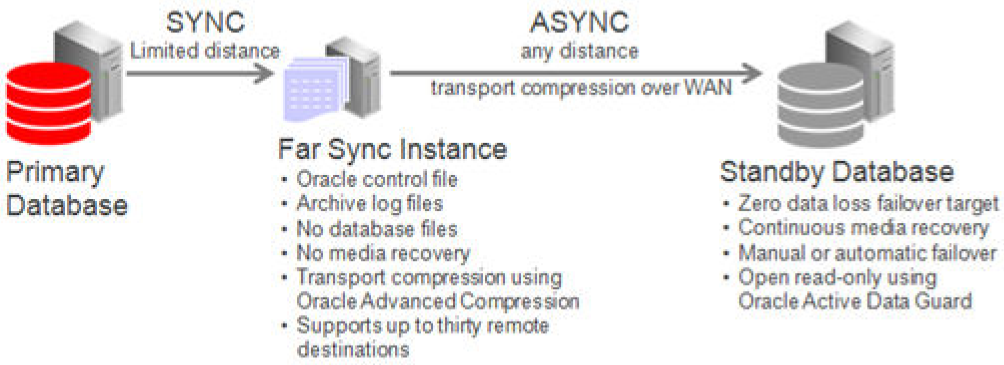
Note that Oracle Advanced Security Option is a licensed option.

Far Sync Deployment Topologies

Oracle Active Data Guard Far Sync provides the ability to perform a zero data loss failover to a remote standby database without requiring a second standby database or complex operation.

Data Guard enables this by deploying a far sync instance (a lightweight Oracle instance that has only a control file, `SPFILE`, password file and standby log files; there are no database files or online redo logs) at a distance that is within an acceptable range of the primary for synchronous transport. A far sync instance receives redo from the primary through synchronous transport and immediately forwards the redo to up to 29 remote standby databases using asynchronous transport. A far sync instance can also forward redo to the new Oracle Database Backup, Logging, and Recovery Appliance.

Figure 12-1   Far Sync Architecture Overview


SYNC
Limited distance
ASYNC
any distance
transport compression over WAN
Primary Database
Far Sync Instance
• Oracle control file
• Archive log files
• No database files
• No media recovery
• Transport compression using Oracle Advanced Compression
• Supports up to thirty remote destinations
Standby Database
• Zero data loss failover target
• Continuous media recovery
• Manual or automatic failover
• Open read-only using Oracle Active Data Guard

The following use cases illustrate the benefits of various architecture choices you can implement with far sync instances.

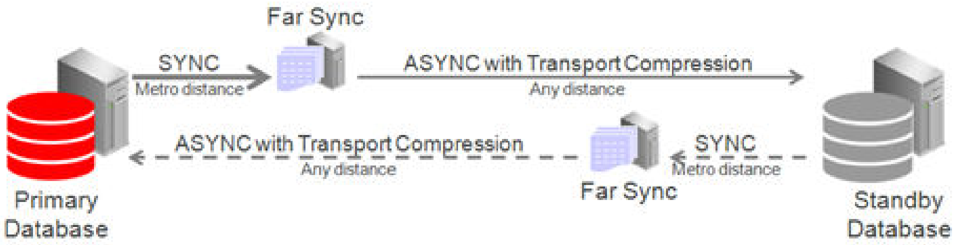
Case 1: Zero Data Loss Protection Following Role Transitions

This is the most basic example in which a primary database uses high availability far sync instances to extend zero data loss failover to a remote standby database.

Ideally the high availability far sync instance is deployed in a location separate from the primary database to isolate it from site failure, but within a metro area distance (network RTT of 5ms or less – subject to performance testing). Even if no separate location is available there is still a benefit to deploying a far sync instance within the same data center to enable fast, zero data loss failover for all unrecoverable outages short of full site failure.

The remote high availability far sync instance is idle while the standby database is in a standby role. It becomes active when the standby database transitions to the primary database role, enabling zero data loss failover to the new standby (old primary). The high availability far sync instance that is local to the original primary database becomes inactive while it is in a standby role.

Figure 12-2   Role Transition Facilitated by Far Sync



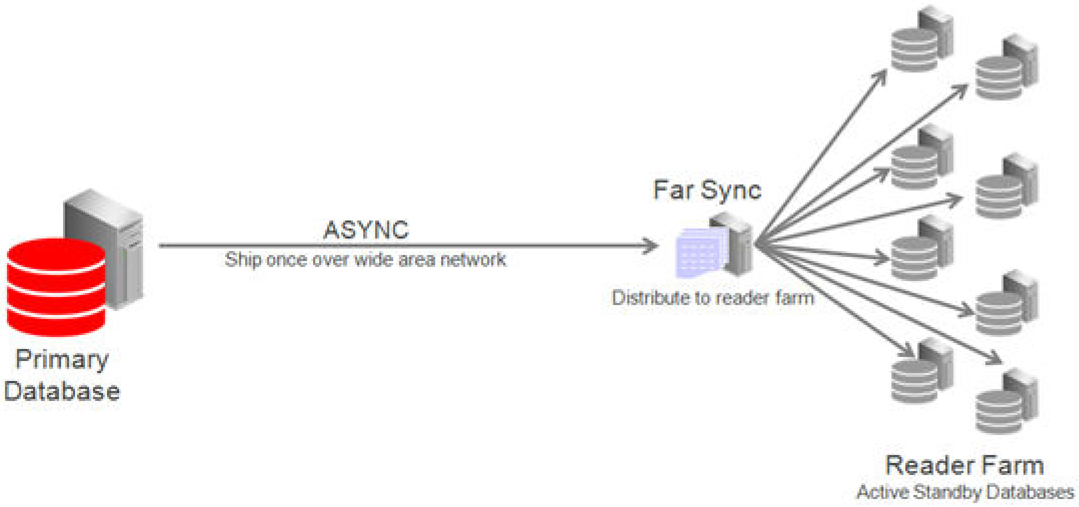
High availability far sync options are described in Far Sync Instance High Availability Typologies.

Case 2: Reader Farm Support

Far Sync can support up to 30 remote destinations, making it a very useful tool for supporting a reader farm – an Active Data Guard configuration having multiple active standby databases to easily scale read performance.

In this example the reader farm is configured in a remote location from the primary database. The primary ships once over the WAN to the far sync instance located in the remote destination and Far Sync distributes redo locally to all active standby databases in the reader farm.

Figure 12-3 Far Sync Ships Redo to Reader Farm



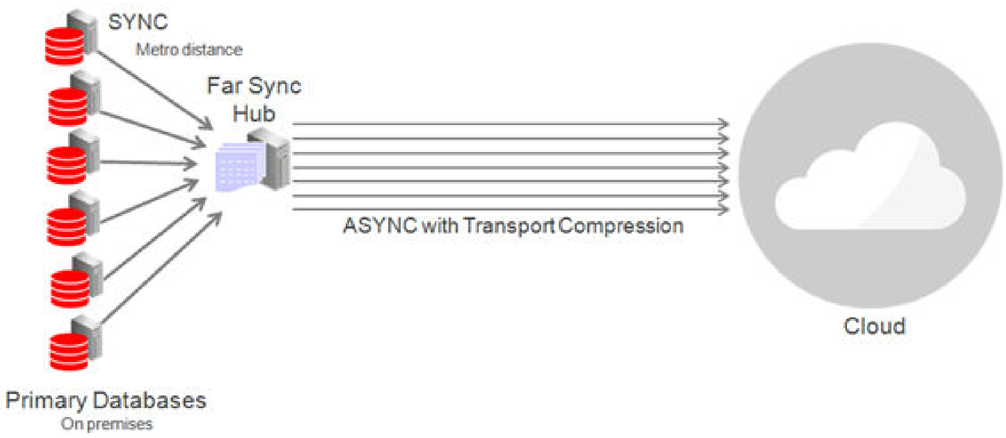
Case 3: Cloud Deployment With Far Sync Hub

Far Sync is a very lightweight process; a single physical server can support multiple far sync instances, each providing zero data loss failover to a remote destination.

The diagram below shows primary databases shipping to a single physical machine operating as a far sync "hub" consisting of multiple far sync instances on a single physical machine. Primaries and the far sync hub are on-premises while standby databases are deployed remotely in the cloud.

Note that all of the systems in this configuration (primary and standby database hosts and far sync instance host) must meet the usual requirements for compatibility in a Data Guard configuration described in Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration (Doc ID 413484.1).

Figure 12-4   Far Sync Hub Architecture



Far Sync High Availability Topologies

To keep far sync instances highly available, consider the following deployment topologies.

Deploy Far Sync Instances on Oracle Real Application Clusters

The far sync instance can be placed on an Oracle RAC cluster. In this configuration a far sync instance is only active on one server at a time while other servers provide automatic failover for high availability. The characteristics of this approach include:

- Lowest data loss potential and brown-out when the active far sync instance or node fails.
- The ability to resume zero data loss protection quickly after far sync instance failure.
- By itself, this solution does not address cluster failure.

The most critical applications are well served by a pair of Oracle RAC far sync instances, each configured as an alternate for the other and deployed at different locations. This provides the most robust HA and data protection (during instance, node, cluster and site outages).

Deploy Far Sync Instances on Alternate Destinations and Multiple Far Sync instances

Configuring two separate far sync instances on distinct physical machines, each serving as an alternate destination for the other, provides far sync instance high availability in a non-Oracle RAC environment. Each destination defined on the primary database contains the ALTERNATE keyword assigning the other far sync instance as the alternate. When the active far sync instance enters an error state the alternate destination pointing to the alternate far sync instance is enabled automatically. By defining a far sync instance as an alternate destination, Maximum Availability protection will be maintained after a briefly dropping to a resynchronization state while the new destination is prepared.

The characteristics of this approach include:

- Retains zero data loss coverage after far sync instance transport failures (instance or network outages).
- Failure testing has shown
  - During far sync instance failures a performance brownout of approximately 3.5 seconds while SYNC redo transport starts (network sync service - NSS).
  - During network failures a short brownout equal to the setting of the destination's net_timeout parameter was observed.
- HA for machine outage assuming each far sync instance is on separate hardware.
- HA for site outage assuming far sync instances are deployed in separate sites.
- Higher application brown-out and resynchronization time during far sync instance outages compared with Far Sync with Oracle RAC

Deploy a Far Sync Instance on the Terminal Standby as an Alternate Destination

The simplest approach to maintaining data protection during a far sync instance outage is to create an alternate `LOG_ARCHIVE_DEST_`*n* pointing directly to the terminal standby (the terminal failover target). Asynchronous transport to the remote destination is the most likely choice in order to avoid the performance impact on the primary caused by WAN network latency.

Asynchronous transport can achieve near-zero data loss protection (as little as sub-seconds to seconds of exposure), but because it never waits for standby acknowledgment, it is unable to provide a zero data loss guarantee. In this configuration the protection level must be dropped to Maximum Performance prior to a switchover (planned event) as the level must be enforceable on the target in order to perform the transition. Changing protection levels and transport methods is a dynamic operation that does not require downtime.

During a far sync instance outage, redo transport automatically fails over to using the alternate destination. Once the far sync instance is repaired and resumes operation, transport automatically switches back to the far sync instance and zero data loss protection is restored.

The characteristics of this approach include:

- No additional hardware or far sync instances to manage.
- Loss of zero data loss coverage during a far sync instance outage. Data protection level drops to `UNSYNCHRONIZED` with `ASYNC` transport until the Far sync instance can resume operation and the standby become fully synchronized.

Choosing a Far Sync Deployment Topology

All configurations for far sync instance high availability perform equally with regard to receiving and sending redo. The choice of configuration should be based on application tolerance to the maximum data loss (RPO) and application brownout period of the different failure scenarios.

- Far sync instances deployed on Oracle RAC provides the lowest brownout and best protection however has no coverage for cluster or site outage. The most critical applications are well served by a pair of Oracle RAC far sync instances configured as alternates for each other and deployed at different locations. This provides the most robust Far Sync high availability (instance, node, cluster, and site failure) protection.
- Alternate far sync instances in a non-RAC environment provide the ability to place each instance on separate physical database servers. This configuration provides protection by deploying the far sync instances in different sites. Applications where data protection is critical but where cost is an important consideration are best served by deploying a pair

of single node far sync instances, each as an alternate for the other. There is, however, slightly increased application brownout and longer resynchronization time while transport transitions from one far sync instance to the other. There is also the potential for data loss should a second outage impact the primary database while transport transitions from one far sync instance to the other.

- Terminal standby alternate configurations require that the application accept that there is no zero data loss protection while the far sync instance is not available, but requires no additional hardware to implement. Applications that can tolerate increased data loss potential during a far sync instance outage, and where low cost is the main consideration, are best served by configuring the terminal standby as an alternate location using asynchronous redo transport. Use of the terminal standby as an alternate destination requires accepting that the configuration will run in asynchronous mode during the entire period required to resolve the far sync instance outage. The advantage of this approach is that it requires no additional hardware or software to deploy or manage. Applications that can tolerate increased data loss potential during a far sync instance outage and where low cost is the main consideration are best served by configuring the terminal standby as an alternate location using ASYNC redo transport.
- A Far Sync hub is an efficient way of consolidating far sync instances for multiple Data Guard configurations on a single physical host. Cloud deployments that include a zero data loss service level category can deploy a Far Sync hub to efficiently consolidate far sync instances for multiple zero data loss configuration on a single physical machine or cluster
- Applications where data protection is critical but where cost is an important consideration are best served by deploying a pair of single node far sync instances, each as an alternate for the other.

Far Sync Configuration Best Practices

The following are far sync configuration best practices that are necessary in addition to those best practices that apply to any synchronous redo transport destination.

- The network between the primary database and the far sync instance must:
  - Have round trip latency low enough so that the impact to response time and throughput of the primary database does not exceed business requirements. The degree of impact is very application specific and will require testing to validate. In general, experience shows that there is a higher likelihood of success if the round-trip latency is less than 5ms, though there are successful deployments at higher latencies.
  - Provide enough bandwidth between the primary database and the far sync instance to accommodate peak redo volumes, in addition to any other traffic sharing the network. Redo transport compression can be used to reduce network bandwidth requirements.
  - Ideally have redundant network links that also tolerate network component failure.
- Standard Oracle Data Guard network best practices, such as setting appropriate TCP send and receive buffer sizes equal to three times the bandwidth delay product. See Configure Online Redo Logs Appropriately.
- Standby redo logs for a far sync instance should be placed on storage with sufficient IOPS (writes per second) capacity to exceed the I/O of the LGWR

process on the primary database during peak activity, in addition to any IOPS from other activities. This is an important consideration. For example:

  - If the far sync instance has lower performing disks than the primary database, it will not be able to forward redo to remote destinations as fast as it is received, and an archive log gap may form.
  - In redo gap resolution scenarios, due to planned maintenance on the standby or network outages, for example, there will be additional I/O requests for gap resolution on top of peak redo coming in.
  - Lower performing disks at the far sync instance will delay acknowledgment to the primary database, increasing the total round-trip time between primary and standby databases and impacting application response time. This impact can be eliminated by using Fast Sync between the primary database and the far sync instance.

- The far sync instance should follow the same standby redo log best practices as the standby database. See Configure Online Redo Logs Appropriately.
- The standby redo logs of an alternate far sync instance should be manually cleared before use to achieve the fastest return to synchronous transport when the alternate far sync is activated. For example:

  ```
  ALTER DATABASE CLEAR LOGFILE GROUP 4, GROUP 5, GROUP 6;
  ```

- Oracle MAA performance testing shows that a small far sync instance SGA does not impact the performance of the far sync instance or the primary database. To conserve system resources, you can configure the minimum SGA required for Far Sync to function.
  - Set `CPU_COUNT=4`. Values of 1 or 2 are possible when neither compression nor encryption are not being used.
  - Reducing the `CPU_COUNT` during testing has no effect on the performance of the Far sync instance.
- Configure far sync instances for both the primary and standby databases to maintain zero data loss protection following role transitions. The second far sync instance configured in proximity to the standby database is idle until the standby becomes the primary database, enabling synchronous redo transport in the reverse direction.

  Note that in a Data Guard Broker configuration, a switchover (planned role transition) cannot occur while in Maximum Availability mode unless the protection mode can be enforced from the target standby site. If the standby database does not have its own far sync instance it will have to be configured to ship asynchronous redo to the original primary database after the roles are reversed. This prevents a switchover from occurring unless the protection mode for the primary database is first dropped from Maximum Availability to Maximum Performance.
- Fast Sync yields a 4% to 12% primary database performance improvement compared to synchronous transport, depending on the network latency and the I/O speed of the far sync instance hardware.
- Provided CPU, I/O, and network requirements are met.
  - Placing the far sync instance on a virtual machine produces no reduction in performance over physical hardware configurations.
  - Multiple far sync instances servicing multiple Data Guard configurations can share the same physical server, cluster, or virtual machine.
- Note that archives may need to be managed on the far sync server.

Configuring the Active Data Guard Far Sync Architecture

The following topics walk you through an example of configuring an Active Data Guard Far Sync architecture.

Configuring the Far Sync Instances

The following examples show you how to add far sync instances to an Oracle Data Guard broker configuration.

The first step is to add a far sync standby instance that is independent or fault isolated from the primary database server, and where the network latency between the primary server and the far sync server is consistently low enough that application performance can tolerate it (for example, < 5 ms).

In the following example, far sync instance FS1 is created for the primary database, North_Sales.

```
DGMGRL> ADD FAR_SYNC FS1 AS CONNECT IDENTIFIER IS FS1.example.com;
Far Sync FS1 added
DGMGRL> ENABLE FAR_SYNC FS1;
Enabled.
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxPerformance
  Members:
  North_Sales  - Primary database
    FS1        - Far Sync
    South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

After a far sync instance has been added to the configuration, set up redo transport to support maximum availability mode and then upgrade the protection mode, as shown in the following example.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'RedoRoutes' =
'(LOCAL : FS1 SYNC)';
DGMGRL> EDIT FAR_SYNC 'FS1' SET PROPERTY 'RedoRoutes' =
'(North_Sales : South_Sales ASYNC)';
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxAvailability
  Members:
  North_Sales  - Primary database
```

```
    FS1          - Far Sync
      South_Sales - Physical standby database

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

To ensure that maximum availability protection mode can be maintained when the remote standby database, South_Sales, becomes the primary database after a switchover or a failover, add a second far sync instance to the configuration so that South_Sales can send redo in synchronous mode, which in turn will send redo to the new terminal database, North_Sales, after the role transition.

The following example shows you how to add a second far sync instance (FS2) to the broker configuration.

```
DGMGRL> ADD FAR_SYNC FS2 AS CONNECT IDENTIFIER IS FS2.example.com;
Far Sync FS2 added
DGMGRL> EDIT FAR_SYNC 'FS2' SET PROPERTY 'RedoRoutes' = '(South_Sales :
North_Sales ASYNC)';
DGMGRL> ENABLE FAR_SYNC FS2;
Enabled.
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'RedoRoutes' = '(LOCAL :
FS2 SYNC)';
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxAvailability
  Members:
  North_Sales  - Primary database
    FS1          - Far Sync
      South_Sales - Physical standby database
      FS2         - Far Sync (inactive)

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

Setting Up HA Far Sync Instances

Alternate HA far sync instances are set up to provide high availability for the far sync instances you created for the primary and remote standby databases.

The following example shows you how to add a second far sync instance (FS1a) to the primary database's far sync instance (FS1) in the Oracle Data Guard broker configuration, so that if the primary far sync instance becomes unavailable, redo transport will use the alternate far sync instance.

```
DGMGRL> ADD FAR_SYNC FS1a AS CONNECT IDENTIFIER IS FS1a.example.com;
Far Sync FS1a added
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'RedoRoutes' = ' (LOCAL:
(FS1 SYNC PRIORITY=1, FS1a SYNC PRIORITY=2))';
DGMGRL> EDIT FAR_SYNC 'FS1' SET PROPERTY 'RedoRoutes' = '(North_Sales :
```

```
South_Sales ASYNC)';
DGMGRL> EDIT FAR_SYNC 'FS1a' SET PROPERTY 'RedoRoutes' =
'(North_Sales : South_Sales ASYNC)';
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxAvailability
  Members:
  North_Sales  - Primary database
    FS1          - Far Sync
    FS1a          - Far Sync
      South_Sales - Physical standby database

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

After adding the alternate far sync instance on the primary, use the following example to add an alternate far sync instance (FS2a) on the standby.

```
DGMGRL> ADD FAR_SYNC FS2a AS CONNECT IDENTIFIER IS FS2a.example.com;
Far Sync FS2a added
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'RedoRoutes' =
' (LOCAL:(FS2 SYNC PRIORITY=1, FS2a SYNC PRIORITY=2))';
DGMGRL> EDIT FAR_SYNC 'FS2' SET PROPERTY 'RedoRoutes' =
'(South_Sales : North_Sales ASYNC)';
DGMGRL> EDIT FAR_SYNC 'FS2a' SET PROPERTY 'RedoRoutes' =
'(South_Sales : North_Sales ASYNC)';
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxAvailability
  Members:
    North_Sales - Primary database
      FS1 - Far Sync
      FS1a - Far Sync
        South_Sales - Physical standby database
          FS2 - Far Sync (inactive)
          FS2a - Far Sync (inactive)

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

Configuring Far Sync Instances with Oracle RAC or Oracle Clusterware

If a far sync instance is deployed on a server or cluster with Oracle Clusterware (for example, in an Oracle Restart, Oracle Real Application Clusters (Oracle RAC), or Oracle RAC One Node installation), then use the SRVCTL utility to specify a default open mode of mount.

You can use a command such as the following:

```
srvctl modify database -d db_unique_name -startoption MOUNT
```

13

Tune and Troubleshoot Oracle Data Guard

When redo transport, redo apply, or role transitions are not meeting your expected requirements, use the following guidelines to help you tune and troubleshoot your deployment.

Overview of Oracle Data Guard Tuning and Troubleshooting

To get the best performance from your Oracle Data Guard configuration, use the following Oracle MAA best practices for monitoring, assessment, and performance tuning.

- Ensure that Oracle Database and Oracle Data Guard configuration best practices are in place.

  The assumption when assessing and tuning is that all of the Oracle Database and Data Guard configuration best practices are already integrated in the environment. Evaluate the adherence to those best practices before doing any tuning.

- Assess and tune redo transport services

  Oracle Data Guard automatically tunes redo transport to optimize performance. However, if you observe performance issues, you can monitor and tune redo transport services.

  Asynchronous redo transport with Maximum Performance data protection mode is the default Oracle Data Guard configuration. Tuning asynchronous redo transport consists mainly of ensuring that the primary, standby, and network resources are sufficient for handling the workload, and that you monitor those resources for bottlenecks.

  Synchronous redo transport does sacrifice some performance for zero data loss; however, using sound MAA recommended methods, you can monitor and assess the impact and distribute resources appropriately.

- Assess and tune redo apply

  In most cases, the default Oracle settings result in satisfactory performance for media recovery when the standby is always up to date. However, as applications and databases increase in size and throughput, media recovery operations can benefit from additional tuning to further optimize recovery time or redo apply throughput on a standby database

- Assess and tune role transitions

  With proper planning and implementation, Oracle Data Guard and Active Data Guard role transitions can effectively minimize downtime and ensure that the database environment is restored with minimal impact on the business. Performance tests using a physical standby database and Oracle Maximum Availability Architecture (MAA) best practices have shown that switchover and failover can be reduced to seconds.

Redo Transport Troubleshooting and Tuning

Oracle Data Guard redo transport performance is directly dependent on the performance of the primary and standby systems, the network that connects them, and the I/O subsystem.

Understanding the topology of the Data Guard configuration and its relevance to Data Guard performance helps eliminate infrastructure weaknesses that are often incorrectly attributed to the Data Guard architecture.

System and Network Performance Prerequisites

Oracle Data Guard architecture is very streamlined and efficient; however, like any application, there are reasonable system and network prerequisites to achieve satisfactory performance. After all of the configuration best practices have been implemented as described in Oracle Data Guard Configuration Best Practices, consider the following recommendations.

Primary Database System

- Sufficient CPU utilization for LGWR to post foregrounds efficiently
- Sufficient I/O bandwidth so local log writes maintain low I/O latency during peak rates
- Network interfaces that can handle peak redo rate volumes combined with any other network activity across the same interface
- Primary AWR, ASH, and OSwatcher data gathered for tuning and troubleshooting

Standby Database System

- Sufficient CPU utilization for RFS (the Data Guard process that receives redo at the standby database) to efficiently write to standby redo logs
- Sufficient I/O bandwidth to enable local log writes to maintain low I/O latency during peak rates
- A network interface that can receive the peak redo rate volumes combined with any other network activity across the same interface
- Standby AWR, ASH, and OSwatcher data should be gathered

> **Note:**
>
> The top problem encountered with the standby database is poor standby log write latency due to insufficient I/O bandwidth. This problem can be mitigated by using Data Guard Fast Sync.

Network

- For synchronous redo transport, the round trip network latency (RTT) between the primary database and target standby database or target far sync standby server should typical less than 2 ms to minimize primary database performance impact.

  Oracle is often asked what the maximum latency is that can be supported or whether there is an equation that can be used to project performance impact. Unfortunately every application has a different tolerance to network latency. Differences in application concurrency, number of sessions, the transaction size in bytes, how often sessions commit, and log switch frequency can cause differences application performance impact.

- Sufficient network bandwidth to support peak redo rates (steady state and when resolving gaps) combined with any other network activity that shares the same network.

  Note that your point to point network bandwidth is throttled by the network segment, switch, router, or interface with the lowest network bandwidth. For example, if you have 10gigE for 90% of your network route and your existing switch or network interface only supports 1 GigE, then your maximum network bandwidth is 1 GigE.
- Netstat and/or any network monitoring statistics should be gathered

> **Note:**
>
> The top network problems encountered are inconsistent network response and insufficient network bandwidth.

Monitor System Resources

Monitor system resources on primary and standby hosts and the network using the utilities and statistics recommended here.

Monitoring CPU

- When all of a system's CPU cores are occupied running work for processes, other processes must wait until a CPU core or thread becomes available, or the scheduler switches a CPU to run their code. A bottleneck can be created if too many processes are queued too often.

  You can use the `uptime`, `mpstat`, `sar`, `dstat`, and `top` utilities to monitor CPU usage.
- The commands `mpstat -P ALL` and `sar -u -P ALL` display CPU usage statistics for each CPU core and an average across all CPU cores.
- The `%idle` value shows the percentage of time that a CPU is not running system code or process code. If the value of `%idle` is near 0% most of the time on all CPU cores, the system is CPU-bound for the workload that it is running. The percentage of time spent running system code (`%systemor %sys`) should not usually exceed 30%, especially if `%idle` is close to 0%.
- The system load average represents the number of processes that are running on CPU cores, waiting to run, or waiting for disk I/O activity to complete, averaged over a period of time. On a busy system, the load average reported by `uptime` or `sar -q` should not exceed two times the number of CPU cores. The system is overloaded if the load average exceeds four times the number of CPU cores for long periods.
- In addition to load averages (`ldavg-*`), the `sar -q` command reports the number of processes currently waiting to run (the run-queue size, `runq-sz`) and the total number of processes (`plist_sz`). The value of `runq-sz` also provides an indication of CPU saturation.
- Determine the system's average load under normal loads when users and applications do not experience problems with system responsiveness, and then look for deviations from this benchmark over time. A dramatic rise in the load average can indicate a serious performance problem.

Monitoring Memory Usage

- When a system runs out of real or physical memory and starts using swap space, its performance deteriorates dramatically. If you run out of swap space, your programs or the entire operating system are likely to crash.

  If `free` or `top` indicate that little swap space remains available, this is also an indication you are running low on memory.

- The `sar -r` command reports memory utilization statistics, including `%memused`, which is the percentage of physical memory in use.
- The `sar -B` command reports memory paging statistics, including `pgscank/s`, which is the number of memory pages scanned by the `kswapd` daemon per second, and `pgscand/s`, which is the number of memory pages scanned directly per second.
- The `sar -W` command reports swapping statistics, including `pswpin/s` and `pswpout/s`, which are the numbers of pages per second swapped in and out per second.
- The system has a memory shortage if `%memused` is near 100% and the scan rate is continuously over 200 pages per second.
- The output from the `dmesg` command might include notification of any problems with physical memory that were detected at boot time.

Monitoring I/O

- Use `iostat`, `sar`, and `iotop` utilities to monitor I/O.
- The `iostat` command monitors the loading of block I/O devices by observing the time that the devices are active relative to the average data transfer rates. You can use this information to adjust the system configuration to balance the I/O loading across disks and host adapters.

  `iostat -x` reports extended statistics about block I/O activity at one second intervals, including `%util`, which is the percentage of CPU time spent handling I/O requests to a device, and `avgqu-sz`, which is the average queue length of I/O requests that were issued to that device.

  If `%util` approaches 100% or `avgqu-sz` is greater than 1, device saturation is occurring and the storage I/O bandwidth needs to be augmented by adding disks or storage.

- You can also use the `sar -d` command to report on block I/O activity, including values for `%util` and `avgqu-sz`.
- The `iotop` utility can help you identify which processes are responsible for excessive disk I/O. The `iotop` user interface is similar to `top`.

  In its upper section, `iotop` displays the total disk input and output usage in bytes per second. In its lower section, iotop displays I/O information for each process, including disk input-output usage in bytes per second, the percentage of time spent swapping in pages from disk or waiting on I/O, and the command name.

  Use the left and right arrow keys to change the sort field, and press A to toggle the I/O units between bytes per second and total number of bytes, or O to toggle between displaying all processes or only those processes that are performing I/O.

Monitoring the Network

- Use `ip`, `ss`, and `sar` to monitor the network.

- The `ip -s` link command displays network statistics and errors for all network devices, including the numbers of bytes transmitted (`TX`) and received (`RX`). The dropped and overrun fields provide an indicator of network interface saturation.
- The `ss -s` command displays summary statistics for each protocol.
- Use the `sar -n DEV` command to monitor the current rate transmitted using an interface.

Assess Database Wait Events

Once you verify that the system or network resources are not bottlenecked, you can assess database wait events for performance flags related to your Oracle Data Guard configuration.

On the primary database, assess database wait events with AWR reports, and on the standby database you can use standby AWR for Oracle Database 18c and later, or `statspack` reports for older Oracle Database releases.

Table 13-1 Wait Events Relevant to Oracle Data Guard

Event Name	Description
ARCH Remote Write	The time (in centi-seconds) that ARCn background processes spend blocked waiting for network write operations to complete
ASYNC Remote Write	The time (in centi-seconds) for asynchronous streaming RFSWRITE operations This includes stall reaps and streaming network submission time. This time is accumulated by TTnn (Redo Transport Slave) background processes.
Redo Transport Attach	The time spent (in centi-seconds) doing Connect, Logon, and RFSATTACH for any network process
Redo Transport Close	The time spent (in centi-seconds) by ARCn, NSSn, and TTnn processes doing RFSCLOSE and RFSRGSTR operations
Redo Transport Detach	The time spent (in centi-seconds) doing RFSDETACH and Disconnect for any network process
Redo Transport Open	The time spent (in centi-seconds) by ARCn, NSSn, and TTnn background processes doing RFSCREAT and RFSANNCE operations
Redo Transport Ping	The time spent (in centi-seconds) by ARCn background processes doing RFSPING operations
Redo Transport Slave Shutdown	The time spent (in centi-seconds) by LGWR doing NSSn and TTnn process shutdown and termination
Redo Transport Slave Startup	The time spent (in centi-seconds) by LGWR doing NSSn and TTnn process startup and initialization
Redo Writer Remote Sync Complete	The time spent (in centi-seconds) by LGWR reaping completed network writes to remote destinations
Redo Writer Remote Sync Notify	The time spent (in centi-seconds) by LGWR issuing network writes to remote destinations

Table 13-1 (Cont.) Wait Events Relevant to Oracle Data Guard

Event Name	Description
Remote SYNC Ping	The time spent (in centi-seconds) by the LGWR and NSSn background processes doing synchronous PING operations
SYNC Remote Write	The time spent by LGWR doing SYNC RFSWRITE operations

Wait events specific to Oracle Data Guard with Oracle Database 11.2 are described in the table below.

Table 13-2 Wait Events Relevant to Oracle Data Guard 11.2

Event Name	Description
ARCH wait on ATTACH	Monitors the amount of time spent by all archiver processes to spawn a new RFS connection
ARCH wait on SENDREQ	Monitors the amount of time spent by all archiver processes to write archive logs to the local disk as well as write them remotely
ARCH wait on DETACH	Monitors the amount of time spent by all archiver processes to delete an RFS connection
LNS wait on ATTACH	Monitors the amount of time spent by all LNS processes to spawn a new RFS connection
LNS wait on SENDREQ	Monitors the amount of time spent by all LNS processes to write the received redo to disk as well as open and close the remote archived redo logs
LNS wait on DETACH	Monitors the amount of time spent by all LNS processes to delete an RFS connection
LGWR wait on LNS	Monitors the amount of time spent by the log writer (LGWR) process waiting to receive messages from LNS processes
LNS wait on LGWR	Monitors the amount of time spent by LNS processes waiting to receive messages from the log writer (LGWR) process
LGWR-LNS wait on channel	Monitors the amount of time spent by the log writer (LGWR) process or the LNS processes waiting to receive messages

See Installing and Using Standby Statspack (Doc ID 454848.1) for information about Standby Statspack.

Assess Synchronous Redo Transport

The following topics describe how to assess redo transport, specifically for synchronous redo transport.

Understanding How Synchronous Transport Ensures Data Integrity

The following algorithms ensure data consistency in an Oracle Data Guard synchronous configuration.

- LGWR redo write on the primary database and the Data Guard NSS network redo write are identical.
- The Data Guard Managed Recovery Process (MRP) at the standby database cannot apply redo unless the redo has been written to the primary's online redo log, with the only exception being during a Data Guard failover operation (primary is gone).

  In addition to shipping redo synchronously, NSS and LGWR exchange information regarding the safe redo block boundary that standby recovery can apply up to from its standby redo logs. This prevents the standby from applying redo it may have received, but which the primary has not yet acknowledged as committed to its own online redo logs.

The possible failure scenarios include:

- If primary database's LGWR cannot write to online redo log, then LGWR and instance will crash. Instance or crash recovery will recover to the last committed transaction in the online redo log and roll back any uncommitted transactions. The current log will be completed and archived.
- On the standby, the partial standby redo log completes with the correct value for the size to match the corresponding online redo log. If any redo blocks are missing from the SRL, those are shipped over (without reshipping the entire redo log).
- If the primary database crashes resulting in an automatic or manual zero data loss failover, then part of the Data Guard failover operation will do "terminal recovery" and read and recover the current standby redo log.

  Once recovery finishes applying all of the redo in the standby redo logs, the new primary database comes up and archives the newly completed log group. All new and existing standby databases discard any redo in the online redo logs, flashback to consistent SCN, and only apply the archives coming from the new primary database. Once again the Data Guard environment is in sync with the (new) primary database.

Assessing Performance in a Synchronous Redo Transport Environment

When assessing performance in a synchronous redo transport environment it is important that you know how the different wait events relate to each other. The impact of enabling synchronous redo transport varies between applications.

To understand why, consider the following description of work the log writer process (LGWR) performs when a commit is issued.

1. Foreground process posts LGWR for commit ("log file sync" starts). If there are concurrent commit requests queued, LGWR will batch all outstanding commit requests together resulting in a continuous strand of redo.
2. LGWR waits for CPU.
3. LGWR starts redo write ("redo write time" starts).
4. For Oracle RAC, LGWR broadcasts the current write to other instances.
5. After preprocessing, if there is a `SYNC` standby, LGWR starts the remote write (“SYNC remote write” starts).

6. LGWR issues local write ("log file parallel write").
7. If there is a SYNC standby, LGWR waits for the remote write to complete.
8. After checking the I/O status, LGWR ends "redo write time / SYNC remote write".
9. For Oracle RAC, LGWR waits for the broadcast ack.
10. LGWR updates the on-disk SCN.
11. LGWR posts the foregrounds.
12. Foregrounds wait for CPU.
13. Foregrounds ends "log file sync".

Use the following approaches to assess performance.

- For batch loads the most important factor is to monitor the elapsed time, because most of these processes must be completed in a fixed period of time. The database workloads for these operations are very different than the normal OLTP workloads. For example, the size of the writes can be significantly larger, so using log file sync averages does not give you an accurate view or comparison.
- For OLTP workloads, monitor the volume of transactions per second (from AWR) and the redo rate (redo size per second) from the AWR report. This information gives you a clear picture of the application throughput and how it is impacted by enabling synchronous redo transport.

Why the log file sync Wait Event is Misleading

Typically, the `log file sync` wait event on the primary database is the first place administrators look when they want to assess the impact of enabling synchronous redo transport.

If the average `log file sync` wait before enabling synchronous redo transport (`SYNC`) was 3ms, and after was 6ms, then the assumption is that `SYNC` impacted performance by one hundred percent. Oracle does not recommend using `log file sync` wait times to measure the impact of `SYNC` because the averages can be very deceiving, and the actual impact of `SYNC` on response time and throughput may be much lower than the event indicates.

When a user session commits, LGWR will go through the process of getting on the CPU, submitting the I/O, waiting for the I/O to complete, and then getting back on the CPU to post foreground processes that their commit has completed. This whole time period is covered by the `log file sync` wait event. While LGWR is performing its work there are, in most cases, other sessions committing that must wait for LGWR to finish before processing their commits. The size and number of sessions waiting are determined by how many sessions an application has and how frequently those sessions commit. This batching up of commits is generally referred to as *application concurrency*.

For example, assume that it normally takes 0.5ms to perform log writes (`log file parallel write`), 1ms to service commits (`log file sync`), and on average you are servicing 100 sessions for each commit. If there was an anomaly in the storage tier, and the log write I/O for one commit took 20ms to complete, then you could have up to 2,000 sessions waiting on `log file sync`, while there would only be 1 long wait attributed to `log file parallel write`. Having a large number of sessions waiting on one long outlier can greatly skew the `log file sync` averages.

The output from V$EVENT_HISTOGRAM for the log file sync wait event for a particular period in time is shown in the following table.

Table 13-3 V$EVENT_HISTOGRAM Output for the Log File Sync Wait Event

Milliseconds	Number of Waits	Percent of Total Waits
1	17610	21.83%
2	43670	54.14%
4	8394	10.41%
8	4072	5.05%
16	4344	5.39%
32	2109	2.61%
64	460	0.57%
128	6	0.01%

The output shows that 92% of the log file sync wait times are less than 8ms, with the vast majority less than 4ms (86%). Waits over 8ms are outliers and only make up 8% of wait times overall, but because of the number of sessions waiting on those outliers (because of batching of commits) the averages get skewed. The skewed averages are misleading when log file sync average waits times are used as a metric for assessing the impact of SYNC.

Understanding What Causes Outliers

Any disruption to the I/O on the primary or standby, or spikes in network latency, can cause high log file sync outliers with synchronous redo transport. You can see this effect when the standby system's I/O subsystem is inferior to that of the primary system.

Often administrators host multiple databases such as development and test on standby systems, which can impair I/O response. It is important to monitor I/O using iostat as described in "Monitoring I/O" in Monitor System Resources to determine if the disks reach maximum IOPS, because this affects the performance of SYNC writes.

Frequent log switches are significant cause of outliers. Consider what occurs on the standby when a log switch on the primary occurs, as follows.

1. RFS on the standby must finish updates to the standby redo log header.
2. RFS then switches into a new standby redo log with additional header updates.
3. Switching logs forces a full checkpoint on the standby. This causes all dirty buffers in the buffer cache to be written to disk, causing a spike in write I/O. In a non-symmetric configuration where the standby storage subsystem does not have the same performance as the primary database, this results in higher I/O latencies.
4. The previous standby redo log must be archived, increasing both read and write I/O.

Effects of Synchronous Redo Transport Remote Writes

When you enable synchronous redo transport (SYNC), you introduce a remote write (RFS write to a standby redo log) in addition to the normal local write for commit processing.

This remote write, depending on network latency and remote I/O bandwidth, can make commit processing time increase. Because commit processing takes longer, you observe more sessions waiting on LGWR to finish its work and begin work on the commit request, that is, application concurrency has increased. You can observe increased application concurrency by analyzing database statistics and wait events. Consider the example in the following table.

Table 13-4 Affect of Sync Transport Increasing Application Concurrency

SYNC	Redo Rate	Network Latency	TPS from AWR	log file sync average (ms)	log file parallel write average (ms)	RFS random I/O	SYNC remote write average (ms)	Redo write size (KB)	Redo writes
Defer	25MB	0	5,514.94	0.74	0.47	NA	NA	10.58	2,246,356
Yes	25MB	0	5,280.20	2.6	.51	.65	.95	20.50	989,791
Impact	0	-	-4%	+251%	+8.5%	NA	NA	+93.8%	-55.9%

In the above example, enabling `SYNC` reduced the number of redo writes, but increased the size of each redo write. Because the size of the redo write increased, you can expect the time spent doing the I/O (both local and remote) to increase. The `log file sync` wait time is higher because there is more work per wait.

However, at the application level, the impact on the transaction rate or the transaction response time might change very little as more sessions are serviced per commit. This is why it is important to measure the impact of `SYNC` at the application level, and not depend entirely on database wait events. It is also a perfect example of why the `log file sync` wait event is a misleading indicator of the actual impact `SYNC` has on the application.

Example of Synchronous Redo Transport Performance Troubleshooting

To look at synchronous redo transport performance, calculate the time spent for local redo writes latency, average redo write size per write, and overall redo write latency, as shown here.

Use the following wait events to do the calculations.

- local redo write latency = 'log file parallel write'
- remote write latency = ‘SYNC remote write’
- average redo write size per write = ‘redo size’ / ‘redo writes’
- average commit latency seen by foregrounds = 'log file sync'

Statistics from an AWR report on an Oracle database are provided in the following table. Synchronous redo transport (`SYNC`) was enabled to a local standby with a 1ms network latency to compare the performance impact to a baseline with `SYNC` disabled.

Table 13-5 Assessing Synchronous Redo Transport Performance with Oracle Database

Metric	Baseline (No SYNC)	SYNC	Impact
redo rate (MB/s)	25	25	no change
log file sync	0.68	4.60	+576%
log file parallel write average (ms)	0.57	0.62	+8.8%
TPS	7,814.92	6224.03	-20.3%

Table 13-5 (Cont.) Assessing Synchronous Redo Transport Performance with Oracle Database

Metric	Baseline (No SYNC)	SYNC	Impact
RFS random I/O	NA	2.89	NA
SYNC remote write average (ms)	NA	3.45	NA
redo writes	2,312,366	897,751	-61,2%
redo write size (KB)	10.58	20.50	+93.8%

In the above example observe that `log file sync` waits averages increased dramatically after enabling `SYNC`. While the local writes remained fairly constant, the biggest factor in increasing `log file sync` was the addition of the `SYNC` remote write. Of the `SYNC` remote write the network latency is zero, so focusing on the remote write into the standby redo log shows an average time of 2.89ms. This is an immediate red flag given that the primary and standby were using the same hardware, and the `SYNC` remote write average time should be similar to the primary's `log file parallel write` average time.

In the above example the standby redo logs have multiple members, and they are placed in a slower performing disk group. After reducing the standby redo logs to a single member and placing them in a fast disk group the you can see results such as those shown in the following table.

Table 13-6 SYNC Performance After Reducing Standby Redo Logs to a Single Member and Placing on a Fast Disk Group

Metric	Baseline (No SYNC)	SYNC	Impact
redo rate (MB/s)	25	25	no change
log file sync	0.67	1.60	+139%
log file parallel write	0.51	0.63	+23.5%
TPS	7714.36	7458.08	-3.3%
RFS random I/O	NA	.89	NA
SYNC remote write average (ms)	NA	1.45	NA
redo writes	2,364,388	996,532	-57.9%
redo write size (KB)	10.61	20.32	+91.5%

Redo Apply Troubleshooting and Tuning

Redo apply performance depends mainly on the type of workload that is being recovered and the system resources allocated to recovery.

Recovering an OLTP workload can be very I/O intensive because there are a large number of small random reads and writes. The higher the number of IOPS (I/O per second) that a storage subsystem can handle, the faster the recovery rate can be.

In contrast, recovering batch workloads is more efficient because it consists of large sequential reads and writes, resulting in much faster recovery rates than OLTP workloads running on equivalent system resources. In addition, batch direct load operation recovery optimizations result in greater efficiency and even higher recovery rates.

The difference between OLTP and batch recovery performance profiles explains why applications with variation in their mixtures of OLTP and batch workloads can have different recovery rates at a standby database, even if the primary database redo generation rates are similar.

As changes occur on the primary database, redo is generated and sent to the standby database. The frequency of shipping redo to the standby is determined by whether the remote destination is using synchronous or asynchronous redo transport.

If redo apply was started using real-time apply, redo generated by the primary database is applied to the standby database as soon as it is received (that is, there is no wait for the database to switch logs). An Oracle Active Data Guard standby database that is open read-only while recovery is active enables users to query current data.

If your standby database is lagging relative to the primary database, you should focus on the following main areas:

- Determining the transport lag
- Determining the apply lag
- Determining the query SCN or time
- Determining how far behind the standby data is compared to the primary database

Monitor Apply Lag

For standby databases on symmetric hardware and configuration, the apply lag should less than 10 seconds.

If you observe high redo transport lag, tune the network transport first and address any insufficient network bandwidth or any overhead resulting from redo encryption or compression.

To monitor apply lag on the standby database query the `V$DATAGUARD_STATS` view.

```
SQL> SELECT name,value,time_computed,datum_time FROM v$dataguard_stats
WHERE name=’%lag’;
```

The `DATUM_TIME` column is the local time on the standby database when the datum used to compute the metric was received. The lag metrics are computed based on data that is periodically received from the primary database. An unchanging value in this column across multiple queries indicates that the standby database is not receiving data from the primary database. The potential data loss in this scenario would be from the last datum time from `V$DATAGUARD_STATS` to the current time on the standby.

To obtain a histogram that shows the history of apply lag values since the standby instance was last started, query the `V$STANDBY_EVENT_HISTOGRAM` view.

```
SQL> SELECT * FROM v$standby_event_histogram WHERE name like '%lag'
and count >0;
```

To evaluate the apply lag over a period of time, take a snapshot of V$STANDBY_EVENT_HISTOGRAM at the beginning of the time period and compare that snapshot with one taken at the end of the time period.

```
SQL> col NAME format a10
SQL> SELECT NAME,TIME,UNIT,COUNT,LAST_TIME_UPDATED FROM
V$STANDBY_EVENT_HISTOGRAM WHERE name like '%lag' and count >0 ORDER BY
LAST_TIME_UPDATED;

NAME              TIME UNIT                  COUNT LAST_TIME_UPDATED
---------- ---------- ---------------- ---------- --------------------
apply lag          41 seconds                   3 04/05/2018 16:30:59
apply lag          44 seconds                   1 04/05/2018 16:31:02
apply lag          45 seconds                   2 04/05/2018 16:31:03
apply lag          46 seconds                   2 04/05/2018 16:31:04
```

Evaluate Redo Apply Rate if the Apply Lag Is High

In an Oracle Data Guard physical standby environment, it is important to determine if the standby database can recover redo as fast as, or faster than, the primary database can generate redo.

If the apply lag is above expectations, then evaluate redo apply performance by querying the V$RECOVERY_PROGRESS view. This view contains the columns described in the following table.

Table 13-7 V$RECOVERY_PROGRESS View Columns

Column	Description
Average Apply Rate	Redo Applied / Elapsed Time includes time spent actively applying redo and time spent waiting for redo to arrive.
Active Apply Rate	Redo Applied / Active Time is a moving average over the last 3 minutes. The rate does not include time spent waiting for redo to arrive.
Maximum Apply Rate	Redo Applied / Active Time is peak measured throughput or maximum rate achieved over a moving average over last 3 minutes. The rate does not include time spent waiting for redo to arrive.
Redo Applied	Represents the total amount of data in bytes that has been applied.
Last Applied Redo	SCN and Timestamp of last redo applied. This is the time as stored in the redo stream, so it can be used to compare where the standby database is relative to the primary.
Apply Time per Log	Average time spent actively applying redo in a log file.
Checkpoint Time per Log	Average time spent for a log boundary checkpoint.
Active Time	Represents the total duration applying the redo , but not waiting for redo
Elapsed Time	Represents the total duration applying the redo , including waiting for redo
Standby Apply Lag	Represents Redo Apply has not applied for N seconds, possible standby is behind the primary.

Table 13-7 (Cont.) V$RECOVERY_PROGRESS View Columns

Column	Description
Log Files	Represents Number of Log files applied so far.

The most useful statistic is the Active Apply rate because the Average Apply Rate includes idle time spent waiting for redo to arrive making it less indicative of apply performance.

The simplest way to determine application throughput in terms of redo volume is to collect Automatic Workload Repository (AWR) reports on the primary database during normal and peak workloads, and determine the number of bytes per second of redo data the production database is producing. You can then compare the speed at which redo is being generated with the Active Apply Rate columns in the `V$RECOVERY_PROGRESS` view to determine if the standby database is able to maintain the pace.

Tune Redo Apply by Evaluating Database Wait Events

Once you have verified that you are not bottlenecked on any system or network resources you are ready to assess database wait events. On the primary database this is done using AWR reports while on the standby database you will use standby AWR.

Before assessing database wait events, it is important that you understand the process flow involved in recovery. In general there are three distinct phases to standby recovery; the log read phase, the apply phase, and the checkpoint phase.

1. Redo is received on the standby by the RFS (Remote File Server) process.

   The RFS process writes newly received redo for each thread into the current standby redo log for that thread. The RFS write operation is tracked by the `rfs random I/O` wait event.

2. Once redo has been written the recovery coordinator process (pr00) will read the redo from the standby redo logs for each thread.

   This read I/O is tracked by the `log file sequential read` operation. The recovery coordinator then merges redo from all threads together and place the redo into memory buffers for the recovery slaves. The wait events for writing and reading into recovery memory buffers is tracked by the `parallel recovery read buffer free` and `parallel recovery change buffer free` wait events.

3. The recovery processes retrieve redo or change vectors from the memory buffers and begin the process applying the changes to data blocks.

   First the recovery slaves determine which data blocks need to be recovered and reads those into the buffer cache if it's not already present. This read I/O by the recovery slaves is tracked by the `recovery read` wait event.

4. When a log is switched on the primary for any thread the standby will coordinate a switch of the standby redo log for that thread at the same time.

   A log switch on a standby will force a full checkpoint which will result in flushing all dirty buffers from the buffer cache out to the data files on the standby. A checkpoint on the standby is currently more expensive than on a primary. Multiple DB writer processes (DBWR) will write the data file blocks down to the data files with its write time tracked by the `db file parallel write` wait event. The total time for the checkpoint to complete is covered by the `checkpoint complete` wait event.

During the apply phase it is normal to observe that the recovery coordinator process (pr00) has high utilization on a single CPU, while during the checkpoint phase you normally see an increase in the write I/O to the data files.

The following table provides a description as well as tuning advice for wait events involved in the recovery process.

Table 13-8 Recovery Process Wait Events

Column	Description	Tuning Recommendations
Logfile sequential read	The parallel recovery coordinator is waiting on I/O from the online redo log or the archived redo log.	Tune or increase the I/O bandwidth for the ASM disk group where the archive logs or online redo logs reside.
Parallel recovery read buffer free	This event indicates that all read buffers are being used by slaves, and usually indicates that the recovery slaves lag behind the coordinator.	Increase `_log_read_buffers` to max 256
Parallel recovery change buffer free	Redo Applied / Active Time is peak measured throughput or maximum rate achieved over a moving average over last 3 minutes. The rate does not include time spent waiting for redo to arrive.	Tune or increase the I/O bandwidth for the ASM disk group where data files reside.
Data file init write	The parallel recovery coordinator is waiting for a buffer to be released by a recovery slave. Again, this is a sign the recovery slaves are behind the coordinator.	Tune or increase the I/O bandwidth for the ASM disk group where data files reside.
Parallel recovery control message reply	The parallel recovery coordinator is waiting for a file resize to finish, as would occur with file auto extend.	This is a non-tunable event.
Parallel recovery slave next change	The parallel recovery slave is waiting for a change to be shipped from the coordinator. This is in essence an idle event for the recovery slave. To determine the amount of CPU a recovery slave is using, divide the time spent in this event by the number of slaves started and subtract that value from the total elapsed time. This may be close, because there are some waits involved.	N/A. This is an idle event.
DB File Sequential Read	A parallel recovery slave (or serial recovery process) is waiting for a batch of synchronous data block reads to complete.	Tune or increase the I/O bandwidth for the ASM disk group where data files reside.

Table 13-8 (Cont.) Recovery Process Wait Events

Column	Description	Tuning Recommendations
Checkpoint completed	Recovery is waiting for checkpointing to complete, and Redo Apply is not applying any changes currently.	Tune or increase the I/O bandwidth for the ASM disk group where data files reside. Also, increase the number of `db_writer_processes` until the checkpoint completed wait event is lower than the db file parallel write wait event. Consider also increasing the online log file size on the primary and standby to decrease the number of full checkpoints at log switch boundaries.
Recovery read	A parallel recovery slave is waiting for a batched data block I/O.	Tune or increase the I/O bandwidth for the ASM disk group where data files reside.
Parallel recovery change buffer free (MIRA)	The parallel recovery coordinator is waiting for a change mapping buffer to be released by one of the recovery slaves.	Increase `_change_vector_buffers` to 2 or 4
Recovery apply pending and/or recovery receive buffer free (MIRA)	Recovery apply pending: the time the `logmerger` process waited (in centiseconds) for apply slaves to apply all pending changes up to a certain SCN. Recovery receive buffer free: the time (in centiseconds) spent by the receiver process on instance waiting for apply slaves to apply changes from received buffers so that they can be freed for the next change.	Increase `_mira_num_local_buffers` and `_mira_num_receive_buffers` Note that these parameters use space from the shared pool equal to the sum of their values (in MB) multiplied by the number of apply instances.

See How to Generate AWRs in Active Data Guard Standby Databases (Doc ID 2409808.1) for more information about generating AWRs on the standby database.

Enable Multi-Instance Redo Apply if Required

Under extremely heavy workloads, a single instance redo apply is not sufficient.

If there are insufficient CPU resources on the standby redo apply database server, or if the Recovery Coordinator process (pr00) is CPU bound (by checking OS utilities), then multi-instance redo apply will allow you to scale across Oracle RAC instances.

Multi-instance redo apply greatly improves the scalability of redo apply for Oracle RAC databases. Rather than merging all threads of redo into a single apply process, multiple apply instances divide the threads of redo between them. For example, when two apply nodes are used to apply four threads of redo from the primary, each apply instance will apply two threads.

When the workload is well balanced between all threads of the primary, the scalability of multi-instance redo apply is predictable (2x, 4x and so on). Unbalanced workloads, where one instance does more work than another, get mixed scaling results compared to single instance redo apply.

Redo Apply Performance Tuning Example

The following is an example which applies the redo apply tuning discussed in this document.

The first step to assess performance and determine steps to improve performance is to take standby `statspack` snaps that capture statistics on the standby database.

After generating a standby `statspack` report based off of those snaps the next step is to examine the load profile section.

```
Load Profile                   Total         Per Second
~~~~~~~~~~~~       ------------------ -----------------
      DB time(s):                9.2                 0.0
       DB CPU(s):                0.7                 0.0
 Redo MB applied:           14,497.8                 9.7
   Logical reads:              471.0                 0.3
  Physical reads:        6,869,213.0             4,570.3
 Physical writes:        9,722,929.0             6,469.0
      User calls:              510.0                 0.3
          Parses:            1,192.0                 0.8
     Hard parses:                3.0                 0.0
W/A MB processed:               35.9                 0.0
          Logons:               45.0                 0.0
        Executes:            1,410.0                 0.9
       Rollbacks:                0.0                 0.0
```

From the load profile shown above, you can see the amount of redo recovered during the report period and the amount of redo recovered on a per second basis. In addition you can see the number of physical reads and writes being performed.

Compare the number of reads and writes to the baseline reports that showed acceptable performance. Increased reads could be coming from read-only queries that might not have been seen previously, while increased writes could indicate a change in the type of workload being recovered.

The top five wait events section is the most important section to review. This section details where the majority of the time is spent waiting, as shown below.

```
Top 5 Timed Events                                    Avg
%Total
~~~~~~~~~~~~~~~~~~                                   wait
Call
Event                                        Waits    Time (s)   (ms)
Time
----------------------------------------- ------------ ----------- ------
------
checkpoint completed                         4,261       8,210   1927
42.1
db file parallel write                      20,982       2,658    127
13.6
lreg timer                                     501       1,502   2998
7.7
```

```
free buffer waits                              119,108       1,278
11    6.5
parallel recovery read buffer free               8,046       1,101
137    5.6
```

In the output above the 42% call time is spent waiting for `checkpoint completed`, with the second wait event `db file parallel write` being associated with the `checkpoint completed` wait as it relates to DBWR performing writes from the buffer cache. The `free buffer waits` wait event indicates that the buffer cache is full and recovery slaves reading buffers into the buffer cache are stalled.

This wait event profile indicates that you should increase the number of DBWR processes to help increase the rate that buffers can be flushed from the buffer cache.

Before making any changes, consult the `statspack` report to make note of the recovery `active apply rate` and the time being spent in the apply phase versus the checkpoint phase. After you adjust the number of DBWR processes, compare the subsequent standby `statspack` reports to these numbers to see improvement.

```
Recovery Start Time Item                       Sofar Units   Redo
Timestamp
------------------- ----------------- -------------- -------
------------------
21-Oct-15 08:03:56  Log Files                      6 Files
21-Oct-15 08:03:56  Active Apply Rate         10,809 KB/sec
21-Oct-15 08:03:56  Average Apply Rat         10,708 KB/sec
21-Oct-15 08:03:56  Maximum Apply Rat         80,592 KB/sec
21-Oct-15 08:03:56  Redo Applied              15,111 Megabyt
21-Oct-15 08:03:56  Last Applied Redo              0 SCN+Tim 20-Oct-15
12:48:25
21-Oct-15 08:03:56  Active Time                1,408 Seconds
21-Oct-15 08:03:56  Apply Time per Lo            121 Seconds
21-Oct-15 08:03:56  Checkpoint Time p             14 Seconds
21-Oct-15 08:03:56  Elapsed Time               1,445 Seconds
21-Oct-15 08:03:56  Standby Apply Lag         70,785 Seconds
```

It's important that you make only one change at a time, and obtain new standby `statspack` reports to assess any improvement based on that change. For each change follow the same methodology of assessing the physical reads and writes, the top wait events, and the time spent in the different recovery phases.

Role Transition Assessment and Tuning

With thorough planning, configuration, and tuning, Oracle Data Guard role transitions can effectively minimize downtime and ensure that the database environment is restored with minimal impact on the business.

Using a physical standby database, Oracle MAA testing has determined that switchover and failover times with Oracle Data Guard have been reduced to seconds. The following topics describe best practices for both switchover and failover. While following best practices, switchover times of approximately 34 seconds for Oracle RAC and 19 seconds for a single instance database have been observed.

Validate Database Switchover and Failover Readiness

The Oracle Data Guard broker `VALIDATE DATABASE` command gathers information related to switchover and failover readiness.

The validation verifies that the standby and primary database are reachable and the apply lag is less than `ApplyLagThreshold` for the target database. If these data points are favorable, the command output displays `Ready for Failover: Yes` as shown below. In addition, if redo transport is running, the command output displays `Ready for Switchover: Yes`.

```
DGMGRL> validate database [verbose] database_name

Database Role: Physical standby database
 Primary Database: standby db_unique_name

Ready for Switchover: Yes
 Ready for Failover: Yes (Primary Running)
```

`VALIDATE DATABASE` checks additional information that can impact switchover time and database performance, such as whether the online redo logs have been cleared, number of temporary tablespaces, parameter mismatches between primary and standby, and the status of flashback databases.

In most failover cases the primary database has crashed or become unavailable. The `Ready for Failover` output indicates if the primary database is running when `VALIDATE DATABASE` was issued. This state does not prevent a failover, but it is recommended that you stop the primary database before issuing a failover to avoid a *split-brain* scenario where the configuration has two primary databases. The broker only guarantees split-brain avoidance on failover when Fast-Start Failover is used.

You can also run `VALIDATE DATABASE` periodically as a configuration monitoring tool.

Use the Broker to Initiate Switchover and Failover

Use the Oracle Data Guard broker SWITCHOVER command to initiate switchover, and the FAILOVER command to initiate failover.

As part of a switchover or failover operation the broker does the following.

- Configures redo transport from the new primary database
- Starts redo apply on the new standby database
- Ensures that other standby databases in the broker configuration are viable and receiving redo from the new primary
- Integrates Oracle Clusterware and Global Data Services to ensure the correct services are started after role change

To configure broker to initiate switchover, run

```
DGMGRL> SWITCHOVER TO database_name;
```

To configure broker to initiate failover, run

```
DGMGRL> FAILOVER TO database_name [IMMEDIATE];
```

By default FAILOVER applies all redo that was received before failing over. The IMMEDIATE clause skips the pending redo and fails over immediately.

The SWITCHOVER and FAILOVER commands are idempotent and can be re-issued in the unlikely event of a failed transition.

Optimize Failover Processing

Implement the following best practices to optimize failover processing.

- Enable Flashback Database to reinstate the failed primary databases after a failover operation has completed. Flashback Database facilitates fast point-in-time recovery from failures caused by user error or logical corruption.
- Enable real-time apply, which allows apply services to apply the redo on the standby databases as soon as it is received.
- Consider configuring multiple standby databases to maintain data protection following a failover.
- Use Oracle Managed Files to pre-clear and pre-create online redo logs on the standby database.

  As part of a failover, the standby database must clear its online redo logs before opening as the primary database. The time needed to complete this I/O can add significantly to the overall failover time. With Oracle Managed Files, the standby pre-creates the online redo logs the first time the managed redo process (MRP) is started.

  - Alternatively set the LOG_FILE_NAME_CONVERT parameter. You can also pre-create empty online redo logs by issuing the SQL*Plus ALTER DATABASE CLEAR LOGFILE statement on the standby database.
- Use fast-start failover. If possible, to optimize switchover processing, ensure that the databases are synchronized before the switchover operation. Real-time apply ensures that redo is applied as received and ensures the fastest switchover.

Enable Fast-Start Failover and Leverage Best Practices

Fast-start failover automatically, quickly, and reliably fails over to a designated standby database if the primary database fails, without requiring manual intervention to execute the failover. You can use fast-start failover only in an Oracle Data Guard configuration that is managed by Oracle Data Guard broker.

The Data Guard configuration can be running in either the maximum availability or maximum performance mode with fast-start failover. When fast-start failover is enabled, the broker ensures fast-start failover is possible only when the configured data loss guarantee can be upheld. Maximum availability mode provides an automatic failover environment guaranteed to lose no data. Maximum performance mode provides an automatic failover environment guaranteed to lose no more than the amount of data (in seconds) specified by the FastStartFailoverLagLimit configuration property.

Adopt fast-start failover best practices discussed in Configure Fast Start Failover.

Assess Time Management Interface Event Alerts to Troubleshoot Role Transition Timings

The Time Management Interface (TMI) event is a low overhead event which adds a line to the alert log whenever certain calls are executed in Oracle.

These entries in the alert log, or *tags*, delineate the beginning and end of a call. The tables in the topics below depict the delineation of key switchover and failover operations. This method is the most accurate for determining where time is being spent.

Set the database level event 16453 trace name context forever, level 15 on all databases. There are two methods of enabling this trace, either using the `EVENT` database parameter or setting the `EVENTS` at the system level. The difference is that the `EVENT` parameter is not dynamic but is persistent across restarts. `set EVENTS` is dynamic but NOT persistent across database restarts. See the following examples.

```
ALTER SYSTEM SET EVENT=‘16453 trace name contextforever, level 15’
scope=spfile sid=’*’


ALTER SYSTEM SET EVENTS ‘16453 trace name context forever, level 15’;
```

Key Switchover Operations and Alert Log Tags

Switchover is broken down into four main steps as follows.

1. `Convert to Standby` - kill any existing production sessions, convert the control file into a standby control file, and send a message to the standby to continue the switchover.

   The `Convert to Standby` - these steps are found in the alert log of the original primary. All remaining steps are found in the original standby alert log.
2. `Cancel Recovery` - apply remaining redo and stop recovery.
3. `Convert to Primary` - a two-step close (to the mounted state) of instances (one instance, then all others), clear online redo logs, convert control file to primary control file, and data Guard Broker bookkeeping.
4. `Open New Primary` - parallel open of all instances.

Table 13-9 Alert Log Tags Defining the Steps with Time Management Interface Event Enabled

Step	Stage	Time Management Interface Event Enabled
Convert To Standby(primary alert log)	BEGIN	TMI: dbsdrv switchover to target BEGIN <DATE> <TIMESTAMP>
Convert To Standby(primary alert log)	END	TMI: kcv_switchover_to_target send 'switchover to primary' msg BEGIN <DATE> <TIMESTAMP>

Table 13-9 (Cont.) Alert Log Tags Defining the Steps with Time Management Interface Event Enabled

Step	Stage	Time Management Interface Event Enabled
Cancel Recovery(standby alert log)	BEGIN	TMI: kcv_commit_to_so_to_primary wait for MRP to die BEGIN <DATE> <TIMESTAMP>
Cancel Recovery(standby alert log)	END	TMI: kcv_commit_to_so_to_primary wait for MRP to die END <DATE> <TIMESTAMP>
Convert to Primary (standby alert log)	BEGIN	TMI: kcv_commit_to_so_to_primary BEGIN CTSO to primary <DATE> <TIMESTAMP>
Convert to Primary (standby alert log)	END	TMI: adbdrv BEGIN 10 <DATE> <TIMESTAMP>
Open Primary(standby alert log)	BEGIN	TMI: adbdrv BEGIN 10 <DATE> <TIMESTAMP>
Open Primary(standby alert log)	END	TMI: adbdrv END 10 <DATE> <TIMESTAMP>

Key Failover Operations and Alert Log Tags

All failover steps are documented in the alert log of the target standby where the failover was performed.

1. Cancel Recovery - Stop recovery and close all instances (to mounted) in parallel.
2. Terminal Recovery - Archive standby redo logs and recover any unapplied redo.
3. Convert to Primary - Clear online redo logs and convert control file to standby control file.
4. Open Primary - Open all instances in parallel.

Table 13-10 Failover Alert Log Tags Defining the Steps with Time Management Interface Event Enabled

Step	Stage	Time Management Interface Event Enabled
Cancel Recovery	BEGIN	TMI: adbdrv termRecovery BEGIN <DATE> <TIMESTAMP>
Cancel Recovery	END	TMI: adbdrv termRecovery END <DATE> <TIMESTAMP>
Terminal Recovery	BEGIN	TMI: krdsmr full BEGIN Starting media recovery <DATE> <TIMESTAMP>
Terminal Recovery	END	TMI: krdemr full END end media recovery <DATE> <TIMESTAMP>

Table 13-10 (Cont.) Failover Alert Log Tags Defining the Steps with Time Management Interface Event Enabled

Step	Stage	Time Management Interface Event Enabled
Convert to Primary	BEGIN	TMI: kcv_commit_to_so_to_primary BEGIN CTSO to primary <DATE> <TIMESTAMP>
Convert to Primary	END	TMI: adbdrv BEGIN 10 <DATE> <TIMESTAMP>
Open Primary	BEGIN	TMI: adbdrv BEGIN 10 <DATE> <TIMESTAMP>
Open Primary	END	TMI: adbdrv END 10 <DATE> <TIMESTAMP>

14
Monitor an Oracle Data Guard Configuration

Use the following Oracle MAA best practice recommendations to monitor an Oracle Data Guard configuration.

Monitoring Oracle Data Guard Configuration Health Using the Broker

The Oracle data Guard broker issues a health check once a minute and updates the configuration status. To force a health check to occur immediately, run the command `show configuration verbose`.

On a primary database, the health check determines if the following conditions are met.

- Database is in the state specified by the user, as recorded in the broker configuration file
- Database is in the correct data protection mode
- Database is using a server parameter file(`SPFILE`)
- Database is in the `ARCHIVELOG` mode
- Redo transport services do not have any errors
- Database settings match those specified by the broker configurable properties
- Redo transport settings match those specified by the redo transport-related properties of the standby databases
- Current data protection level is consistent with configured data protection mode
- Primary database is able to resolve all gaps for all standby databases

On a standby database, the health check determines whether the following conditions are met.

- Database is in the state specified by the user, as recorded in the broker configuration file
- Database is using a server parameter file (`SPFILE`)
- Database settings match those specified by the broker configurable properties
- Primary and target standby databases are synchronized or within lag limits if fast-start failover is enabled

To identify any warnings on the overall configuration, show the status using the `SHOW CONFIGURATION` command.

```
DGMGRL> show configuration;

Configuration – dg

  Protection Mode: MaxPerformance
  Members:
```

```
  tin - Primary database
    can - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS   (status updated 18 seconds ago)
```

If the configuration status is `SUCCESS`, everything in the broker configuration is working properly.

However, if you see a status of `WARNING` or `ERROR`, then something is wrong in the configuration. Additional error messages will accompany the `WARNING` or `ERROR` status that should be used to identify current issues.

The next step is to examine each database in the configuration to narrow down what the specific error is related to.

To identify the warnings on the primary database, get its status using the `SHOW DATABASE` command.

```
DGMGRL> show database tin

Database - tin

  Role:               PRIMARY
  Intended State:     TRANSPORT-ON
  Instance(s):
    tin1
    tin2

Database Status:
SUCCESS
```

If the database status is `SUCCESS` then the database is working properly.

However, if you see a status of `WARNING` or `ERROR`, then something is wrong in the database. Additional error messages will accompany the `WARNING` or `ERROR` status that should be used to identify current issues.

Repeat the same `SHOW DATABASE` command on the standby database and assess any error messages.

In addition to the above commands, the broker features a `VALIDATE DATABASE` command.

```
DGMGRL> validate database tin

  Database Role:    Primary database
  Ready for Switchover:  Yes

DGMGRL> validate database can;

  Database Role:     Physical standby database
  Primary Database:  tin
  Ready for Switchover:  No
  Ready for Failover:    Yes (Primary Running)

  Capacity Information:
    Database  Instances        Threads
    tin       2                2
    can       1                2
    Warning: the target standby has fewer instances than the
    primary database, this may impact application performance

  Standby Apply-Related Information:
    Apply State:      Not Running
    Apply Lag:        Unknown
    Apply Delay:      0 minutes
```

The `VALIDATE DATABASE` does not provide a `SUCCESS` or `WARNING` status and must be examined to determine if any action needs to be taken.

It is recommended that you run the `VALIDATE DATABASE` command after creating the broker configuration, and before and after any role transition operation.

The `VALIDATE DATABASE` command performs the following checks.

- Whether there is missing redo data on a standby database
- Whether flashback is enabled
- The number of temporary tablespace files configured
- Whether an online data file move is in progress
- Whether online redo logs are cleared for a physical standby database
- Whether standby redo logs are cleared for a primary database
- The online log file configuration
- The standby log file configuration
- Apply-related property settings
- Transport-related property settings
- Whether there are any errors in the Automatic Diagnostic Repository (for example, control file corruptions, system data file problems, user data file problems)

Detecting Transport or Apply Lag Using the Oracle Data Guard Broker

Given enough resources, in particular network bandwidth, an Oracle Data Guard standby can maintain pace with very high workloads. In cases where resources are constrained, the standby can begin to fall behind, resulting in a transport or apply lag.

A **transport lag** is the amount of data, measured in time, that the standby has not received from the primary.

An **apply lag** is the difference, in elapsed time, between when the last applied change became visible on the standby and when that same change was first visible on the primary.

When using the Data Guard broker, the transport or apply lag can be viewed by using the `SHOW DATABASE` command and referencing the standby database, as shown here.

```
DGMGRL> show database orclsb

Database - orclsb

  Role:               PHYSICAL STANDBY
  Intended State:     APPLY-ON
  Transport Lag:      0 seconds (computed 0 seconds ago)
  Apply Lag:          0 seconds (computed 1 second ago)
  Average Apply Rate: 792.00 KByte/s
  Real Time Query:    ON
  Instance(s):
    orclsb1 (apply instance)
    orclsb2

Database Status:
SUCCESS
```

The broker `TransportDisconnectedThreshold` database property (default of 0 in Oracle Database 11.2, and 30 seconds for Oracle Database 12.1 and later releases) can be used to generate a warning status for a standby when the last communication from the primary database exceeds the value specified by the property. The property value is expressed in seconds.

The following is an example of the warning when a disconnection has occurred.

```
DGMGRL> show database orclsb;

Database - orclsb

  Role:               PHYSICAL STANDBY
  Intended State:     APPLY-ON
  Transport Lag:      0 seconds (computed 981 seconds ago)
  Apply Lag:          0 seconds (computed 981 seconds ago)
  Average Apply Rate: 12.00 KByte/s
  Real Time Query:    OFF
  Instance(s):
    orclsb1 (apply instance)
    orclsb2

  Database Warning(s):
    ORA-16857: member disconnected from redo source for longer than 
specified threshold
```

The broker also has the following configurable database properties that you can use to generate warnings when a transport or apply lag exceed a user defined value.

- The `ApplyLagThreshold` property generates a warning status for a logical or physical standby when the database's apply lag exceeds the value specified by the property.

The property value is expressed in seconds. A value of 0 seconds results in no warnings being generated when an apply lag exists. As a best practice, Oracle recommends setting `ApplyLagThreshold` to at least 15 minutes.

- The `TransportLagThreshold` property can be used to generate a warning status for a logical, physical, or snapshot standby when the database's transport lag exceeds the value specified by the property.

  The property value is expressed in seconds. A value of 0 seconds results in no warnings being generated when a transport lag exists. As a best practice, Oracle recommends setting `TransportLagThreshold` to at least 15 minutes.

Monitoring Oracle Data Guard Configuration Health Using SQL

You can use the queries in the following tables to assess the overall Data Guard configuration health on the primary database and the standby database.

Table 14-1 Primary Database Queries

Goal	Query	Expected Results
Check if any remote standby archive destination is getting errors Check if all remote standby archive destinations is enabled or `VALID`	`select sysdate,status,error from gv$archive_dest_status where type='PHYSICAL' and status!='VALID' or error is not null;`	Good health = no rows returned If the query returns rows, then raise an alert with the returned data.
Check if any `NOLOGGING` activity occurred on the primary database in the last day	`select file#, name, unrecoverable_change#, unrecoverable_time from v$datafile where unrecoverable_time > (sysdate - 1);`	Good health = no rows returned If the query returns rows, then the standby database is vulnerable, and the files listed in the output must be refreshed on the standby.
Detect gaps on the standby database	`select sysdate,database_mode,recovery_mode, gap_status from v$archive_dest_status where type='PHYSICAL' and gap_status !='NO GAP';`	Good health = no rows returned If the query returns rows, then there's an existing gap between the primary and the standby database, and you must run the same query on the standby database. If the output from the primary and standby is identical, then no action is required. If the output on the standby does not match the output from the primary, then the datafile on the standby should be refreshed.

Table 14-1 (Cont.) Primary Database Queries

Goal	Query	Expected Results

Assess whether any severe Data Guard event occurred in the last day	`select *`
	` from v$dataguard_status`
	` where severity in ('Error','Fatal')`
	` and timestamp > (sysdate -1);`
		Good health = no rows returned
		If the query returns rows, then raise an alert with the returned output.

FOR SYNC ENVIRONMENTS ONLY:
Assess if running in Maximum Availability mode and configuration is in sync
	`select sysdate,protection_mode,`
	` synchronized, synchronization_status`
	` from v$archive_dest_status`
	` where type='PHYSICAL'`
	` and synchronization_status !='OK';`
		Good health = no rows returned
		If the query returns rows, then raise an alert with the returned output.

Table 14-2 Physical Standby Database Queries

Goal	Query	Expected Results

Determine if there is a transport lag	`select name,value,time_computed,datum_time`
	` from v$dataguard_stats`
	` where name='transport lag'`
	` and value > '+00 00:01:00';`
		Good health = no rows returned
		If no rows are returned, then this implies that there is no transport lag

Determine if there is an apply lag	`select name,value,time_computed,datum_time`
	` from v$dataguard_stats`
	` where name='apply lag'`
	` and value > '+00 00:01:00';`
		Good health = no rows returned
		If no rows are returned, then this implies that there is no apply lag

Table 14-2 (Cont.) Physical Standby Database Queries

Goal	Query	Expected Results

Standby data file check (offline files or files that are not accessible)	`select *`
	` from v$datafile_header`
	` where status ='OFFLINE'`
	` or ERROR is not null;`
		Good health = no rows returned
		Any rows returned list the files that have I/O or recovery issues

Verify that the Media Recovery Process is currently running	`select *`
	` from v$managed_standby`
	` where process like 'MRP%';`
		Good health = rows returned
		If no rows are returned, then the MRP process is not running

Assess whether any severe Data Guard event occurred in the last day	`select *`
	` from v$dataguard_status`
	` where severity in`
	`('Error','Fatal')`
	` and timestamp > (sysdate`
	`-1);`
		Good health = no rows returned
		If the query returns rows, then raise an alert with the returned output

Oracle Data Guard Broker Diagnostic Information

The Oracle Data Guard broker provides information about its activities in several forms.

- Database status information
- Oracle alert log files

  The broker records key information in the alert log file for each instance of each database in a broker configuration.
- Oracle Data Guard broker log files

  For each instance of each database in a broker configuration, the broker DMON process records important behavior and status information in a broker log file, which are useful for diagnosing Oracle Data Guard failures. The Set the `TraceLevel` configuration property to specify the level of diagnostic information reported in the broker log files. The broker log file is created in the same directory as the alert log and is named `drc<$ORACLE_SID>.log`.
- Oracle Data Guard command line (DGMGRL) `logfile` option

  If the DGMGRL command-line interface was started with the `-logfile` optional parameter, then the resulting log file may contain a useful record of past operations and error conditions.

Detecting and Monitoring Data Corruption

If corrupt data is written to disk, or if a component failure causes good data to become corrupt after it is written, then it is critical that you detect the corrupted blocks as soon as possible.

To monitor the database for errors and alerts:

- Query the `V$DATABASE_BLOCK_CORRUPTION` view that is automatically updated when block corruption is detected or repaired.
- Configure Data Recovery Advisor to automatically diagnose data failures, determine and present appropriate repair options, and perform repair operations at your request.

  Note that Data Recovery Advisor integrates with the Oracle Enterprise Manager Support Workbench (Support Workbench), the Health Monitor, and RMAN.
- Use Data Guard to detect physical corruptions and to detect lost writes.

  Data Guard can detect physical corruptions when the apply process stops due to a corrupted block in the redo steam or when it detects a lost write.

  Use Enterprise Manager to manage and monitor your Data Guard configuration.

  By taking advantage of Automatic Block Media Recovery, a corrupt block found on either a primary database or a physical standby database can be fixed automatically when the Active Data Guard option is used.
- Use SQL*Plus to detect data file corruptions and inter-block corruptions.

  Run this SQL*Plus statement:

  ```
  sqlplus> ANALYZE TABLE table_name VALIDATE STRUCTURE CASCADE;
  ```

  After finding the corruptions, the table can be re-created or another action can be taken.
- An Recovery Manager (RMAN) backup and recovery strategy can detect physical block corruptions.
  A more intensive RMAN check using the following command can detect logical block corruptions.

  ```
  RMAN> BACKUP VALIDATE CHECK LOGICAL;
  ```

Part IV

Oracle GoldenGate High Availability Best Practices

15

Overview of Oracle GoldenGate High Availability Best Practices

Oracle GoldenGate is Oracle's strategic logical replication product and integral part of MAA's platinum architecture. By adding Oracle GoldenGate replication, a Gold MAA reference architecture is elevated to a Platinum MAA reference architecture.

The Platinum reference architecture has the potential to provide zero downtime for outages and planned maintenance activities that are not achievable with the Gold architecture. To learn more about the Platinum MAA reference architecture, see High Availability Reference Architectures.

Oracle GoldenGate provides the following benefits:

- Uni-directional or bi-directional replication, allowing reads and updates in any replicated database.
- Data movement is in real-time, reducing latency.
- Replicated databases can run on different hardware platforms, database versions, and different database or application configurations, allowing for online migration. This flexibility also allows online database and application upgrades.
- Source and target replicated databases are online, so zero downtime switch over of applications, during outages and planned maintenance activities is possible. Note, the application switchover must be customized, rather than using a built-in feature, such as Transparent Application Continuity.

The following table highlights various Oracle GoldenGate configuration best practices.

Table 15-1 Oracle GoldenGate Use Cases and Best Practices

Use Case	Oracle GoldenGate Best Practices
Database migration to Oracle Cloud	Zero Downtime Migration (ZDM) with GoldenGate (logical migration) Migration to the Oracle Cloud with an Oracle GoldenGate Hub Configuration
Database migration requiring minimal or zero downtime Database migration involving cross platform or different database versions	Oracle Database Migration with an Oracle GoldenGate Hub Configuration

Table 15-1 (Cont.) Oracle GoldenGate Use Cases and Best Practices

Use Case	Oracle GoldenGate Best Practices
Deploy Oracle GoldenGate off of the database server in a Hub configuration, offering the following benefits: • Offloads Oracle GoldenGate software installation and configuration from database servers to a single server or cluster. • Offloads some of the CPU and IO consumption of Oracle GoldenGate processes. Extract redo mining is still done on the database servers. • Leverages Oracle GoldenGate Microservices Architecture, simplifying Oracle GoldenGate configuration, administration and management.	Oracle Maximum Availability Architecture (MAA) GoldenGate Hub
Optimize Oracle GoldenGate configuration for performance	Oracle GoldenGate Performance Best Practices
Configure Oracle GoldenGate on Real Application Cluster Configure Oracle GoldenGate on Real Application Cluster in Oracle Cloud	Oracle GoldenGate Microservices Architecture with Oracle Real Application Clusters Configuration Best Practices or Oracle GoldenGate With Oracle Real Application Clusters Configuration Oracle GoldenGate Microservices Architecture on Oracle Cloud Infrastructure
Tips to integrate other MAA technologies with Oracle GoldenGate	Oracle Exadata MAA - Platinum Tier Focused Presentation Disaster Recovery for Oracle Database: Zero Data Loss Recovery Appliance, Active Data Guard and Oracle GoldenGate - An Overview Transparent Role Transitions with Oracle Data Guard and Oracle GoldenGate
Application Failover Options for Oracle GoldenGate	Global Data Services Concepts and Administration Guide

Also see Oracle GoldenGate documentation at: https://docs.oracle.com/en/middleware/goldengate/core/21.1/

Part V

Oracle Database Cloud Best Practices

16

Oracle Maximum Availability Architecture and Autonomous Database Cloud

Oracle Maximum Availability Architecture (MAA) is a set of best practices developed by Oracle engineers over many years for the integrated use of Oracle High Availability, data protection, and disaster recovery technologies. The key goal of Oracle MAA is to meet Recovery Time Objectives (RTO) and Recovery Point Objectives (RPO) for Oracle databases and applications running on our system and database platforms using Oracle Cloud MAA architectures and solutions.

See Oracle MAA Reference Architectures for an overview of the MAA reference architectures and their associated benefits and potential RTO and RPO targets. Also, see Oracle Maximum Availability Architecture in Exadata Cloud Database Systems for the inherent differentiated Exadata HA and data protection benefits, since Autonomous Database Cloud runs on the Exadata Cloud platform.

Note that Maximum Availability Architectures leverage Chaos Engineering throughout their testing and development life cycles to ensure that end-to-end application and database availability is preserved, or at its optimal levels, for any fault or maintenance event in Oracle Cloud. *Chaos Engineering* is the discipline of experimenting on a system to build confidence in the system’s capability to withstand turbulent conditions in production. Specifically, MAA aggressively injects various faults and planned maintenance events to evaluate application and database impact throughout our development, stress, and testing cycles. With that experimentation, best practices, defects, and lessons learned are derived, and that knowledge is put back into practice to evolve and improve our cloud MAA solutions.

Autonomous Database with Default High Availability Option

High availability is suitable for all development, test, and production databases that have high uptime requirements and zero or low data loss tolerance. By default, Autonomous Databases are highly available, incorporating a multi-node configuration to protect against localized hardware failures.

Each Autonomous Database application service resides in at least one Oracle Real Application Clusters (Oracle RAC) instance, with the option to fail over to another available Oracle RAC instance for unplanned outages or planned maintenance activities, enabling zero or near-zero downtime.

Autonomous Database's automatic backups are stored in Oracle Cloud Infrastructure Object Storage and are replicated to another availability domain. These backups can be used to restore the database in the event of a disaster.

Major database upgrades do require some downtime, although it can be minimized by relocating a pluggable database (PDB) to an upgraded container database (CDB). Note that the ability to relocate a PDB to an upgraded CDB is only available with Autonomous Database with Shared Infrastructure.

The uptime service-level objective per month is 99.95% (a maximum of 22 minutes of downtime per month), but when you use Maximum Availability Architecture best practices for continuous application service, most months would have zero downtime. The uptime service-level objective does not include downtime due to customer-initiated high availability tests, disaster recovery (such as an availability domain or regional outage), database corruptions, or downtime due to planned maintenance that cannot be done online or by using an Oracle RAC rolling update solution, such as major database upgrades from one release to another.

The following table describes the recovery-time objectives and recovery-point objectives (data loss tolerance) for service-level objectives.

Table 16-1 Default High Availability Policy Recovery Time and Recovery Point Service-level Objectives

Failure and Maintenance Events	Database Downtime	Service-level Downtime (RTO)	Potential Service-level Data Loss (RPO)
Localized events, including: • Exadata cluster network topology failures • Storage (disk and flash) failures • Database instance failures • Database server failures • Periodic software and hardware maintenance updates	Zero	Near-zero	Zero
Events that require restoring from backup because the standby database does not exist: • Data corruptions • Full database failures • Complete storage failures • Availability domain or region failures	Minutes to hours (without Autonomous Data Guard)	Minutes to hours (without Autonomous Data Guard)	15 minutes (without Autonomous Data Guard)

In the table above, the amount of downtime for events that require restoring from a backup varies depending on the nature of the failure. In the most optimistic case, a physical block corruption is detected and the block is repaired with block media recovery in minutes. In this case, only a small portion of the database is affected with zero data loss. In a more pessimistic case, an availability domain or data region fails, and a new cluster must be provisioned and restored with the latest database backup, including all archives, and a complete database recovery must be run.

Data loss is limited by the last successful archive log backup, the frequency of which is every 15 minutes by default, and includes a log switch and subsequent archive log backup of any redo that has not been backed up to Oracle Cloud Infrastructure Object Storage. Data loss can be seconds, or, at worst, around 15 minutes. For Autonomous Database on Dedicated Infrastructure, archive backup frequency is every 15 minutes. For Autonomous Database on Shared Infrastructure, archive backup frequency is every hour.

Autonomous Database with Autonomous Data Guard Option

Enable Autonomous Data Guard for mission-critical production databases that have stricter uptime requirements than databases with the default high-availability configuration and limited data-loss tolerance.

Enabling Autonomous Data Guard adds one symmetric standby database with Oracle Data Guard to an Exadata rack that is located in another availability domain or in another region. The primary and standby database systems are configured symmetrically by default to ensure that performance service levels are maintained after Data Guard role transitions.

Oracle Data Guard features asynchronous redo transport (in maximum performance mode) by default to ensure zero application performance impact. Data Guard zero data loss protection can be achieved by changing to synchronous redo transport (in maximum availability mode); however, Maximum Availability database protection mode with synchronous redo transport is only available with Autonomous Database on Dedicated Infrastructure). The standby database can be placed within the same availability domain, across availability domains, or across regions.

As with databases that are not Data Guard-enabled, each Autonomous Database application service resides in at least one Oracle RAC instance and will automatically fail over to another available Oracle RAC instance, as previously described. The read-only standby database provides expanded application services to offload reporting, queries, and some updates. Read-only standby database is only available with Autonomous Data Guard on Dedicated Infrastructure.

The Database Backup Cloud Service schedules automated backups, which are stored in Oracle Cloud Infrastructure Object Storage and replicated to another availability domain. Those backups can be used to restore databases in the event of a double disaster where both primary and standby databases are lost.

Local and remote virtual cloud network peering provides a secure, high-bandwidth network across availability domains and regions for any traffic between primary and standby servers.

The uptime service-level objective per month is 99.995% (maximum 132 seconds of downtime per month) and recovery-time objectives (downtime) and recovery-point objectives (data loss) are low, as described in the table below. When you use Maximum Availability Architecture best practices for application continuous service, most months would have a downtime of zero. The uptime service-level objective does not include downtime as a result of user-initiated high availability tests, user-initiated Data Guard switchover tests, detection time to determine whether the primary database is down, and time leading up to customer initiated manual Data Guard switchover or failover operation.

Automatic Data Guard failover with zero data loss on Shared Infrastructure can be initiated when there’s a deterministic database failure, and the calculated downtime starts at the time the Data Guard failover operation is issued and completes when the standby is converted to the new primary database and is open to service transactions. Automatic Data Guard Failover is only available with Autonomous Data Guard on Shared Infrastructure, and only if zero data loss is guaranteed; otherwise, a manual Data Failover option will be provided in the OCI Console.

You can choose whether your database failover site is located in a different availability domain within the same region or in a different region, contingent upon application or business requirements and data center availability.

Table 16-2 Autonomous Data Guard Recovery Time and Recovery Point Service-level Objectives

Failure and Maintenance Events	Service-level Downtime (RTO)	Potential Service-level Data Loss (RPO)
Localized events, including: • Exadata cluster network fabric failures • Storage (disk and flash) failures • Database instance failures • Database server failures • Periodic software and hardware maintenance updates	Zero or Near Zero	Zero
Events that require failover to the standby database using Autonomous Data Guard, including: • Data corruptions (because Data Guard has automatic block repair for physical corruptions[1], a failover operation is required only for logical corruptions or extensive data corruptions) • Full database failures • Complete storage failures • Availability domain or region failures	Few seconds to two minutes	• Zero with maximum availability protection mode (uses synchronous redo transport). Most commonly used for intra-region standby databases.[2] • Near zero for maximum performance protection mode (uses asynchronous redo transport). Most commonly used for cross-region standby databases.

[1] The Active Data Guard automatic block repair for physical corruptions feature is only available for Autonomous Data Guard on Dedicated Infrastructure.

[2] Active Data Guard Maximum Availability protection mode with synchronous redo transport (zero data loss protection) is only available with Autonomous Data Guard on Dedicated Infrastructure. For complete container database (CDB) failure (the most common disaster) or complete storage and data center failures, zero data loss Data Guard failover is available for Autonomous Data Guard on Dedicated Infrastructure. Autonomous Data Guard on Shared Infrastructure provides automatic zero data loss failover if the source primary system and CDB is available; otherwise, you can do a minimum data loss manual failover using the OCI Console or APIs.

Maintaining Application Uptime

Ensure that network connectivity to Oracle Cloud Infrastructure is reliable so that you can access your tenancy's Autonomous Database resources.

Follow the guidelines to connect to your autonomous database (Shared Infrastructure, Dedicated Infrastructure). Applications must connect to the predefined service name and download client credentials that include the proper `tnsnsames.ora` and `sqlnet.ora` files. You can also change your specific application service's drain_timeout attribute to fit your requirements.

For more details about enabling continuous application service through planned and unplanned outages, see Application Checklist for Continuous Service for MAA Solutions. Oracle recommends that you test your application readiness by following Validating Application Failover Readiness (Doc ID 2758734.1).

Depending on the Exadata cloud planned maintenance event, Oracle attempts to automatically relocate services and drain sessions prior to stopping any Oracle RAC instance. For OLTP applications that follow the MAA checklist, draining and relocating services results in zero application downtime.

Some applications, such as long running batch jobs or reports, may not be able to drain and relocate gracefully, even with a longer drain timeout. For those applications, Oracle recommends that you schedule the software planned maintenance window excluding these types of activities, or stop these activities before the planned maintenance window. For example, you can reschedule a planned maintenance window so that it is outside your batch windows, or stop batch jobs before a planned maintenance window.

17

Oracle Maximum Availability Architecture in Exadata Cloud Database Systems

Oracle Maximum Availability Architecture in Exadata Cloud Service and Oracle Cloud@Customer provides inherent high availability, data protection, and disaster recovery protection integrated with both cloud automation and life cycle operations, enabling Oracle Exadata Cloud Database Systems to be the best cloud solution for enterprise databases and applications.

See Oracle Cloud: Maximum Availability Architecture for detailed walk-through of Oracle Cloud MAA architectures and features.

Oracle Maximum Availability Architecture Benefits

- **Deployment**: Oracle deploys Exadata in Oracle Cloud Infrastructure using Oracle Maximum Availability Architecture best practices, including configuration best practices for storage, network, operating system, Oracle Grid Infrastructure, and Oracle Database. Exadata is optimized to run enterprise Oracle databases with extreme scalability and availability.
- **Oracle Maximum Availability Architecture database templates**: All cloud databases created with Oracle Cloud automation use Oracle Maximum Availability Architecture default settings, which are optimized for Exadata in Oracle Cloud. Oracle does not recommend that you use custom scripts to create cloud databases. Other than adjusting memory and system resource settings, avoid migrating previous database parameter settings, especially undocumented parameters.
- **Backup and restore automation**: When you configure automatic backup to Oracle Cloud Infrastructure Object Storage, backup copies provide additional protection when multiple availability domains exist in your region, and RMAN validates cloud database backups for any physical corruptions. Database backups occur daily, with a full backup occurring once per week and incremental backups occurring on all other days. Archive log backups occur frequently to reduce potential data loss in case of disaster.
- **Exadata inherent benefits**: Exadata is the best Oracle Maximum Availability Architecture platform that Oracle offers. Exadata is engineered with hardware, software, database, and availability innovations that support the most mission-critical enterprise applications. Specifically, Exadata provides unique high availability, data protection, and quality-of-service capabilities that set Oracle apart from any other platform or cloud vendor. Sizing Exadata cloud systems to meet your application and database system resource needs (for example. sufficient CPU, memory, and IO resources) is very important to maintain the highest availability, stability, and performance. Proper sizing is especially important when consolidating many databases on the same cluster.

For a comprehensive list of Oracle Maximum Availability Architecture benefits for Exadata database systems, see Exadata Database Machine: Maximum Availability Architecture Best Practices. Examples of these benefits include:

- **High availability and low brownout**: Fully-redundant, fault-tolerant hardware exists in the storage, network, and database servers. Resilient, highly-available software, such as

Oracle Real Application Clusters (Oracle RAC), Oracle Clusterware, Oracle Database, Oracle Automatic Storage Management, Oracle Linux, and Oracle Exadata Storage Server enable applications to maintain application service levels through unplanned outages and planned maintenance events.

For example, Exadata has instant failure detection that can detect and repair database node, storage server, and network failures in less than two seconds, and resume application and database service uptime and performance. Other platforms can experience 30 seconds, or even minutes, of blackout and extended application brownouts for the same type of failures. Only the Exadata platform offers a wide range of unplanned outage and planned maintenance tests to evaluate end-to-end application and database brownouts and blackouts.

- **Data protection**: Exadata provides Oracle Database with physical and logical block corruption prevention, detection, and, in some cases, automatic remediation. The Exadata Hardware Assisted Resilient Data (HARD) checks include support for server parameter files, control files, log files, Oracle data files, and Oracle Data Guard broker files, when those files are stored in Exadata storage. This intelligent Exadata storage validation stops corrupted data from being written to disk when a HARD check fails, which eliminates a large class of failures that the database industry had previously been unable to prevent.

  Examples of the Exadata HARD checks include:

  - Redo and block checksum
  - Correct log sequence
  - Block type validation
  - Block number validation
  - Oracle data structures, such as block magic number, block size, sequence number, and block header and tail data structures

  Exadata HARD checks are initiated from Exadata storage software (cell services) and work transparently after enabling a database DB_BLOCK_CHECKSUM parameter, which is enabled by default in the cloud. Exadata is the only platform that currently supports the HARD initiative.

  Furthermore, Oracle Exadata Storage Server provides non-intrusive, automatic hard disk scrub and repair. This feature periodically inspects and repairs hard disks during idle time. If bad sectors are detected on a hard disk, then Oracle Exadata Storage Server automatically sends a request to Oracle Automatic Storage Management to repair the bad sectors by reading the data from another mirror copy.

  Finally, Exadata and Oracle Automatic Storage Management can detect corruptions as data blocks are read into the buffer cache, and automatically repair data corruption with a good copy of the data block on a subsequent database write. This inherent intelligent data protection makes Exadata and Exadata Cloud the best data protection storage platform for Oracle Databases.

  For comprehensive data protection a Maximum Availability Architecture best practice is to use a standby database on a separate Exadata instance to detect, prevent, and automatically repair corruptions that cannot be addressed by Exadata alone. The standby database also minimizes downtime and data loss for disasters that result from site, cluster, and database failures.

- **Response time quality of service**: Only Exadata has end-to-end quality-of-service capabilities to ensure that response time remains low and optimum. Database server I/O latency capping and Exadata storage I/O latency capping ensure that read or write I/O can be redirected to partnered cells when response time exceeds a certain threshold. If storage becomes unreliable (but not failed) because of poor and unpredictable performance, then the disk or flash cache can be confined offline, and later brought back online if heuristics show that I/O performance is back to acceptable levels. Resource management can help prioritize key database network or I/O functionality, so that your application and database perform at an optimized level. For example, database log writes get priority over backup requests on Exadata network and storage. Furthermore, rapid response time is maintained during storage software updates by ensuring that partner flash cache is warmed so flash misses are minimized.

- **End-to-end testing and holistic health checks**: Because Oracle owns the entire Exadata Cloud infrastructure, end-to-end testing and optimizations benefit every Exadata customer around the world, whether hosted on-premises or in the cloud. Validated optimizations and fixes required to run any mission-critical system are uniformly applied after rigorous testing.

  Health checks are designed to evaluate the entire stack. The Exadata health check utility EXACHK is Exadata cloud-aware and highlights any configuration and software alerts that may have occurred because of customer changes. No other cloud platform currently has this kind of end-to-end health check available. For Oracle Autonomous Database, EXACHK runs automatically to evaluate Maximum Availability Architecture compliance. For non-autonomous databases, Oracle recommends running EXACHK at least once a month, and before and after any software updates, to evaluate any new best practices and alerts.
- **Higher Uptime**: The uptime service-level agreement per month is 99.95% (a maximum of 22 minutes of downtime per month), but when you use MAA best practices for continuous service, most months would have zero downtime.

Full list of Exadata features and benefits: Whats New in Oracle Exadata Database Machine

Oracle Maximum Availability Architecture best practices paper: Oracle Maximum Availability Architecture (MAA) engineering collaborates with Oracle Cloud teams to integrate Oracle MAA practices that are optimized for Oracle Cloud Infrastructure and security. See MAA Best Practices for the Oracle Cloud for additional information about continuous availability, Oracle Data Guard, Hybrid Data Guard, Oracle GoldenGate, and other Maximum Availability Architecture-related topics.

Expected Impact with Unplanned Outages

The following table lists various unplanned outages and the associated potential database downtime, application level Recovery Time Objective (RTO), and data loss potential or recovery point objective (RPO). For Data Guard architectures, the database downtime or service level downtime does not include detection time or the time it takes before a customer initiates the Cloud Console Data Guard failover operation.

Table 17-1 Availability and Performance Impact for Exadata Cloud Software Updates

Failure and Maintenance Events	Database Downtime	Service-Level Downtime (RTO)	Potential Service-Level Data Loss (RPO)
Localized events, including: Exadata cluster network topology failures Storage (disk and flash) failures Database instance failures Database server failures	Zero	Near-zero	Zero
Events that require restoring from backup because a standby database does not exist: Data corruptions Full database failures Complete storage failures Availability domain or region failures	Minutes to hours (without Data Guard)	Minutes to hours (without Data Guard)	30 minutes (without Data Guard)
Events using Data Guard to fail over: Data corruptions Full database failures Complete storage failures Availability domain or region failures	Seconds to minutes[1] Zero downtime for physical corruptions due to auto-block repair feature	Seconds to minutes The foreground process that detects the physical corruption pauses while auto block repair completes	Zero for Max Availability (SYNC) Near Zero for Max Performance (ASYNC)

[1] Database downtime excludes 1) primary database failure detection time, and 2) all the time up to manually initiating the Data Guard failover or switchover command.

Expected Impact with Planned Maintenance

The following table lists various software updates and the associated database and application impact. This is applicable for all of Oracle's Exadata Cloud services, including Exadata Cloud at Customer, Exadata Cloud Service Gen2, and Autonomous Database Cloud.

Table 17-2 Availability and Performance Impact for Exadata Cloud Software Updates

Software Update	Database Impact	Application Impact	Scheduled By	Performed By
Exadata Network Fabric Switches	Zero downtime with No Database Restart	Zero to single-digit seconds brownout	Customer	Oracle Cloud for both Autonomous Database and non-Autonomous Database

Table 17-2 (Cont.) Availability and Performance Impact for Exadata Cloud Software Updates

Software Update	Database Impact	Application Impact	Scheduled By	Performed By
Exadata Storage Servers	Zero downtime with No Database Restart	Zero to single-digit seconds brownout Exadata storage servers are updated in rolling manner maintaining redundancy Oracle Exadata System Software pre-fetches the secondary mirrors of the OLTP data that is most frequently accessed into the flash cache, maintaining application performance during storage server restarts Exadata smart flash for database buffers is maintained across storage server restart With Exadata 21.2 software, Persistent Storage Index and Persistent Columnar Cache features enable consistent query performance after a storage server software update	Customer	Oracle Cloud for both Autonomous Database and non-Autonomous Database
Exadata Database Host	Zero downtime with Oracle RAC rolling updates	Zero downtime Exadata Database compute resources are reduced until planned maintenance completes	Customer	Oracle Cloud for both Autonomous Database and non-Autonomous Database
Exadata Database Guest	Zero downtime with Oracle RAC rolling updates	Zero downtime Exadata Database compute resources are reduced until planned maintenance completes	Customer for Autonomous Database	Oracle Cloud for Autonomous Database Customer using Oracle Cloud Console/APIs for non-Autonomous Database

Table 17-2 (Cont.) Availability and Performance Impact for Exadata Cloud Software Updates

Software Update	Database Impact	Application Impact	Scheduled By	Performed By
Oracle Database quarterly update or patch	Zero downtime with Oracle RAC rolling updates	Zero downtime Exadata Database compute resources are reduced until planned maintenance completes	Customer for Autonomous Database	Oracle Cloud for Autonomous Database Customer using Oracle Cloud Console/APIs or dbaascli utility for non-Autonomous Database
Oracle Grid Infrastructure quarterly update, patch, or upgrade	Zero downtime with Oracle RAC rolling updates	Zero downtime Exadata Database compute resources are reduced until planned maintenance completes	Customer for Autonomous Database	Oracle Cloud for Autonomous Database Customer using Oracle Cloud Console/APIs or dbaascli utility for non-Autonomous Database
Oracle Database upgrade with downtime	Minutes to Hour(s) downtime	Minutes to Hour(s) downtime	Customer for Autonomous Database	Oracle Cloud for Autonomous Database Customer using Oracle Cloud Console/APIs or dbaascli utility for non-Autonomous Database
Oracle Database upgrade with near zero downtime	Minimal downtime with DBMS_ROLLING, Oracle GoldenGate replication, or with pluggable database relocate	Minimal downtime with DBMS_ROLLING, Oracle GoldenGate replication, or with pluggable database relocate		Oracle Cloud for Autonomous Database on Shared Infrastructure can execute pluggable database relocate for upgrade use cases Customer using generic Maximum Availability Architecture best practices for non-Autonomous Database

Exadata cloud systems have many elastic capabilities that can be used to adjust database and application's performance needs. By rearranging resources on need, you can maximize system resources to targeted databases and applications and you can minimize costs. The following table lists elastic Exadata Infrastructure and VM Cluster updates and the impacts associated with those updates on databases and applications. All of these operations unless specified otherwise can be performed by using Oracle Cloud Console/APIs.

Table 17-3 Availability and Performance Impact for Exadata Elastic Operations

VM Cluster Changes	Database Impact	Application Impact
Scale Up or Down VM Cluster Memory Exadata Cloud@Customer only	Zero downtime with Oracle RAC rolling updates	Zero to single-digit seconds brownout Sufficient database compute memory and hugepages are required to avoid database server instability. Adjusting hugepage settings may be required.
Scale Up or Down VM Cluster CPU Applicable for Exadata Cloud@Customer and Exadata Cloud Service	Zero downtime with No Database Restart	Zero downtime Application performance and throughput can be impacted by available CPU resources
Scale Up or Down (resize) ASM Storage for Database usage Exadata Cloud@Customer only	Zero downtime with No Database Restart	Zero downtime Oracle ASM rebalance is initiated automatically, but redundancy or application performance will not be compromised
Scale Up VM Local File System Size Exadata Cloud@Customer via Oracle Cloud Console. Exadata Cloud Service via Service Request	Zero downtime with No Database Restart for scaling up Zero downtime with Oracle RAC rolling updates for scaling down	Zero downtime
Adding Exadata Storage Cells applicable for Exadata Cloud@Customer and Exadata Cloud Service	Zero downtime with No Database Restart	Zero to single-digit seconds brownout Oracle ASM rebalance is initiated automatically, but redundancy or application performance should not be compromised
Adding Exadata Database Servers Exadata Cloud Service only	Zero downtime with No Database Restart	Zero to single-digit seconds brownout Application performance and throughput can be impacted by available CPU resources
Adding/Dropping Database Nodes in Virtual Machines (VMs) Cluster	Zero downtime with No Database Restart	Zero downtime

Because some of these elastic changes may take significant time and may impact available resources for your application, some planning is required. Note that “scale down” and “drop” changes will decrease available resources. Care must be taken to not reduce resources below the amount required for database and application stability and to meet application performance targets. Refer to the following table for estimated timings and planning recommendations.

Table 17-4 Customer Planning Recommendations for Exadata Elastic Operations

VM Cluster Changes	Estimated Timings	Customer Planning Recommendations
Scale Up or Down VM Cluster Memory Exadata Cloud@Customer only	Time to drain services and Oracle RAC rolling restart Typically 15-30 minutes per node, but may vary depending on application draining	Understanding application draining. Refer to Achieving Continuous Availability for your Applications section. (link) When scaling down memory, ensure that database SGAs can still be stored in hugepages, and that application performance is still acceptable. To preserve predictable application performance and stability: • Monitor and scale up before important high workload patterns require the memory resources • Avoid memory scale down unless all your Databases' SGA and PGA memory fit into the new memory size and that all SGAs are accommodated by system's huge pages.
Scale Up or Down VM Cluster CPU Applicable for Exadata Cloud@Customer and Exadata Cloud Service	Online operation Typically less than 5 minutes per VM cluster. Scaling up from a very low value to very high value (10+ OCPUs increase) may take 10 minutes.	To preserve predictable application performance and stability: • Monitor and scale up before important high workload patterns require the CPU resources or when consistently reaching an OCPU threshold for tolerated amount of time. • Only scale down if the load average is below a threshold for at least 30 minutes or scale down based on fixed workload schedules (e.g. business hours with 60 oCPUs, non-business hours with 10 oCPUs and batch with 100 oCPUs) • Avoid more than one scale down requests within 2 hours period

Table 17-4 (Cont.) Customer Planning Recommendations for Exadata Elastic Operations

VM Cluster Changes	Estimated Timings	Customer Planning Recommendations
Scale Up or Down (resize) ASM Storage for Database usage Exadata Cloud@Customer only	Time varies based on utilized database storage capacity and database activity. The higher percentage of utilized database storage, the longer the resize operation (which includes ASM rebalance) will take. Typically minutes to hours.	Due to inherent best practices of using non-intrusive ASM power limit, application workload impact is minimal. Choose a non-peak window so resize and rebalance operations can be optimized. Since the time may vary significantly, plan for the operation to complete in hours. To estimate the time that an existing resize or rebalance operation per VM cluster, query GV$ASM_OPERATION. For example a customer can run the following query every 30 minutes to evaluate how much work (EST_WORK) and how much more time (EST_MINUTES) potentially is required: `select operation, pass, state, sofar, est_work, est_minutes from gv$asm_operation where operation='REBAL';` Note the estimated statistics tend to become more accurate as the rebalance becomes further along but can vary based on the concurrent workload.
Scale Up VM Local File System Size Exadata Cloud@Customer via Oracle Cloud Console. Exadata Cloud Service via Service Request	Exadata X8M and later: Online operation, typically less than 5 minutes per VM cluster Exadata X8 and earlier: Time to drain services and Oracle RAC rolling restart. Typically 15-30 minutes per node, but may vary depending on application draining settings.	Understanding application draining. See Achieving Continuous Availability For Your Applications
Scale Down VM Local File System Size Exadata Cloud@Customer via Oracle Cloud Console Exadata Cloud Service via Service Request	Time to drain services and Oracle RAC rolling restart Typically 15-30 minutes per node, but may vary depending on application draining	Understanding application draining Refer to Achieving Continuous Availability for your Applications section. (link)

Table 17-4 (Cont.) Customer Planning Recommendations for Exadata Elastic Operations

VM Cluster Changes	Estimated Timings	Customer Planning Recommendations
Adding Exadata Storage Cells applicable for Exadata Cloud@Customer and Exadata Cloud Service	Online operation to create more available space for administrator to choose how to distribute. **Exadata Cloud Service:** Typically 3-72 hours per "add cell" operation depending on database storage usage and storage activity. In some cases it will 3 hours if system is idle. With very active database and heavily storage activity, this can take up to take 72 hours. As part of the add storage cell operation, all disk groups will expanded automatically. **Exadata Cloud@Customer:** Typically 3-72 hours per "add cell(s)" operation depending number of VM clusters, database storage usage and storage activity. As part of the add storage cell operation, there are two parts to this operation. 1) storage is added to the Exadata system as part the add storage, 2) administrator needs to decide which VM cluster to expand its ASM disk groups as a separate operation.	Plan to add storage when your storage capacity utilization will hit 80% within a month's time since the operation may complete in days. Due to inherent best practices of using non-intrusive ASM power limit, application workload impact is minimal. Since the time may vary significantly, plan for the operation to complete in days before the storage is available. To estimate the time that an existing resize or rebalance operation per VM cluster, query GV$ASM_OPERATION. For example a customer can run the following query every 30 minutes to evaluate how much work (EST_WORK) and how much more time (EST_MINUTES) potentially is required: `select operation, pass, state, sofar, est_work, est_minutes from gv$asm_operation where operation='REBAL';` Note the estimated statistics tend to become more accurate as the rebalance becomes further along but can vary based on the concurrent workload.
Adding Exadata Database Servers Exadata Cloud Service only	Online operation to expand your VM cluster. One step process to add the Database Compute to the Exadata infrastructure and then expand the VM cluster. Approximately 1 to 6 hours per Exadata Database Server	Plan to add Database Compute when your Database resource utilization or growth will hit 80% within a month's time. Be aware and plan for this operation to take many hours to a day. Choose a non-peak window so that the add Database Compute operation can complete faster Estimate a half day before database compute is available Each Oracle RAC database registered by Oracle clusterware AND Oracle Cloud will be extended. If a database was configured outside the OCI Cloud console or without dbaascli, then those databases will not be extended.

Table 17-4 (Cont.) Customer Planning Recommendations for Exadata Elastic Operations

VM Cluster Changes	Estimated Timings	Customer Planning Recommendations
Adding/Dropping Database Nodes in Virtual Machines (VMs) Cluster	Zero database downtime when adding Database Nodes in VM cluster typically takes 3-6 hours, depending on the number of databases in the VM cluster Zero database downtime with dropping Database Nodes in VM cluster typically takes 1-2 hours, depending on number of databases in the VM cluster	Application brownout of maximum of 10 seconds because of Oracle RAC cluster reconfiguration Understand that the add/drop operation is not instantaneous, and operation may take several hours to complete Drop operation reduces Database compute, OCPU and memory resources, so application performance can be impacted

Achieving Continuous Availability For Your Applications

As part of Exadata Cloud, all software updates (except for non-rolling database upgrades or non-rolling patches) can be done online or with Oracle RAC rolling updates to achieve continuous database uptime. Furthermore, any local failures of the storage, Exadata network, or Exadata database server are managed automatically, and database uptime is maintained.

To achieve continuous application uptime during Oracle RAC switchover or failover events, follow these application-configuration best practices:

- Use Oracle Clusterware-managed application services to connect your application. For Data Guard environments, use role based services.
- Use recommended connection string with built-in timeouts, retries, and delays, so that incoming connections do not see errors during outages.
- Configure your connections with Fast Application Notification.
- Drain and relocate services. Refer to Table 5, Application Drain Attributes for Exadata Cloud Software Updates. Use recommended best practices that support draining such as test connections when borrowing or starting batches of work, and return connections to pools between uses.
- Leverage Application Continuity or Transparent Application Continuity to replay in-flight uncommitted transactions transparently after failures.

For more details on the above checklist, refer to Application Checklist for Continuous Service for MAA SolutionsOracle recommends testing your application readiness by following Validating Application Failover Readiness (Doc ID 2758734.1).

Depending on the Exadata cloud planned maintenance event, Oracle attempts to automatically drain and relocate application services before stopping any Oracle RAC instance. For OLTP applications, draining and relocating services typically work very well and result in zero application downtime. Some applications, such as long running batch jobs or reports, may not be able to drain and relocate gracefully within the maximum drain timeout. For those applications, Oracle recommends scheduling the software planned maintenance window around these types of activities or stopping these activities before the planned maintenance window. For example, you can reschedule a planned maintenance window to run outside your batch windows, or stop batch jobs before a planned maintenance window.

The following table lists planned maintenance events that perform Oracle RAC instance rolling restart, and the relevant service drain timeout variables that may impact your application.

Table 17-5 Application Drain Attributes

Exadata Cloud Software Updates or Elastic Operations	Drain Timeout Variables
Oracle DBHOME patch apply and database MOVE	Cloud software automation stops each database instance while honoring drain_timeout settings defined by clusterware managed service configuration (e.g. srvctl).[1] You can override drain_timeout defined on services for a specific command line *dbaascli dbHome patch* or *dbaascli database move* operation with option –*drainTimeoutInSeconds.* The Oracle cloud internal maximum draining time supported is 2 hours.
Oracle Grid Infrastructure (GI) patch apply and GI upgrade	Cloud software automation stops each database instance while honoring drain_timeout settings defined by clusterware managed service configuration (e.g. srvctl).1 You can override drain_timeout defined on services for a specific command line *dbaascli grid patch* or *dbaascli gridupgrade* operation with option *–drainTimeoutInSeconds.* The maximum draining time supported is 2 hours.
Virtual Machine Operating System Software Update (Exadata Database Guest)	Exadata patchmgr/dbnodeupdate software program calls drain orchestration (rhphelper). Drain orchestration has the following drain timeout settings (See MOS Note 2385790.1 for details): • DRAIN_TIMEOUT – if a service does not have drain_timeout defined, then this value is used. Default value is 180 seconds. • MAX_DRAIN_TIMEOUT - overrides any higher drain_timeout value defined for a given service, default 300. No max timeout. • Each Clusterware managed service is also controlled by drain_timeout attribute that can be lower than the above values.

Table 17-5 (Cont.) Application Drain Attributes

Exadata Cloud Software Updates or Elastic Operations	Drain Timeout Variables
Exadata X8 and earlier systems • Scale Up and Down VM Local File System Size • Scale Up or Down VM Cluster Memory	Exadata X8 and earlier systems local file system resize operation calls drain orchestration (rhphelper). Drain orchestration has the following drain timeout settings (See MOS Note 2385790.1 for details): • DRAIN_TIMEOUT – if a service does not have drain_timeout defined, then this value is used. Default value is 180 seconds. • MAX_DRAIN_TIMEOUT - overrides any higher drain_timeout value defined for a given service, default 300. • Each Clusterware managed service is also controlled by drain_timeout attribute that can be lower than the above values. • The Oracle cloud internal maximum draining time supported is 300 seconds.
Exadata X8M and later systems • Scale Down VM Local File System Size	Exadata X8M and later systems call drain orchestration (rhphelper). Drain orchestration has the following drain timeout settings (See MOS Note 2385790.1 for details): • DRAIN_TIMEOUT – if a service does not have drain_timeout defined, then this value is used. Default value is 180 seconds. • MAX_DRAIN_TIMEOUT - overrides any higher drain_timeout value defined for a given service, default 300. • Each Clusterware managed service is also controlled by drain_timeout attribute that can be lower than the above values. • The Oracle cloud internal maximum draining time supported is 300 seconds.

Table 17-5 (Cont.) Application Drain Attributes

Exadata Cloud Software Updates or Elastic Operations	Drain Timeout Variables
Exadata X8M and later systems • Scale Up or Down VM Cluster Memory	Exadata X8M and later systems call drain orchestration (rhphelper). Drain orchestration has the following drain timeout settings (See MOS Note 2385790.1 for details): • DRAIN_TIMEOUT – if a service does not have drain_timeout defined, then this value is used. Default value is 180 seconds. • MAX_DRAIN_TIMEOUT - overrides any higher drain_timeout value defined for a given service, default 300. • Each Clusterware managed service is also controlled by drain_timeout attribute that can be lower than the above values. • The Oracle cloud internal maximum draining time supported is 900 seconds.
Exadata Database Host (Infrastructure) software update	Exadata Database Host calls drain orchestration (rhphelper). Drain orchestration has the following drain timeout settings (See MOS Note 2385790.1 for details): • DRAIN_TIMEOUT – if a service does not have drain_timeout defined, then this value is used. Default value is 180 seconds. • MAX_DRAIN_TIMEOUT - overrides any higher drain_timeout value defined for a given service, default 300. • Each Clusterware managed service is also controlled by drain_timeout attribute that can be lower than the above values. • The Oracle cloud internal maximum draining time supported is – For Exadata X8 and earlier systems, the timeout is 300 seconds. – For Exadata X8M and later systems, the timeout is 500 seconds.

[1] Minimum software requirements to get this service drain capability is 1) database version 12.2 and later and 2) latest cloud dbaas tooling software

Oracle Maximum Availability Architecture Reference Architectures in the Exadata Cloud

Exadata Cloud supports all Oracle Maximum Availability Architecture reference architectures, providing support for all Oracle Databases, regardless of their specific high availability, data protection, and disaster recovery service-level agreements. See

MAA Best Practices for the Oracle Cloud for more information about Oracle Maximum Availability Architecture in the Exadata Cloud.