Grid Infrastructure: Overview

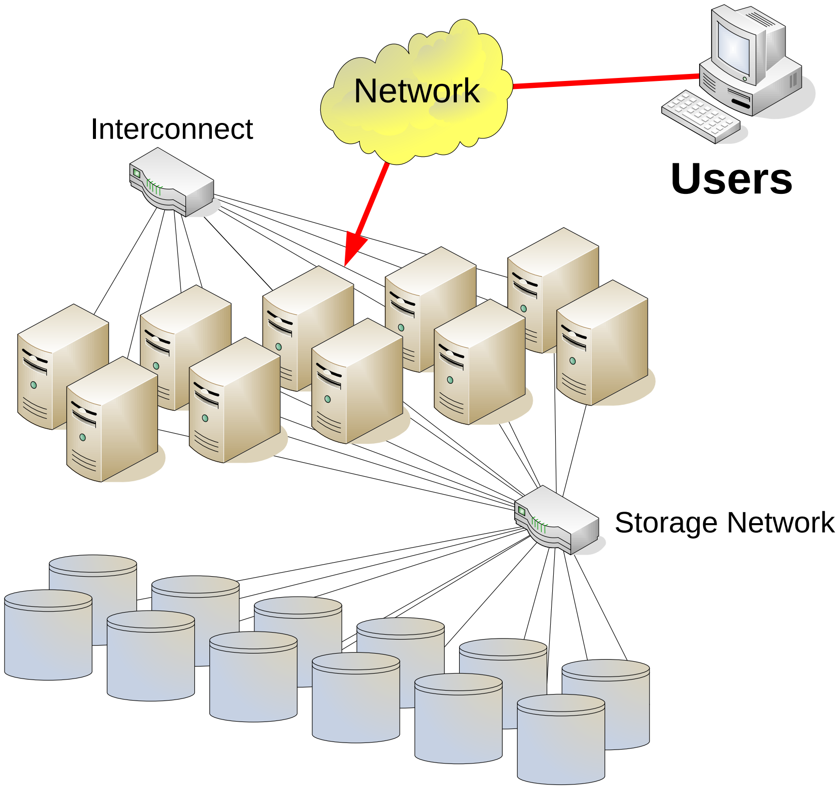
Objectives

After completing this lesson, you should be able to:

- Explain the principles and purposes of clusters
- Describe the Oracle Clusterware architecture
- Describe how Grid Plug and Play affects Clusterware

Cluster

- A group of independent, but interconnected, computers that act as a single system
- Usually deployed to increase availability and performance or
to balance a dynamically changing workload



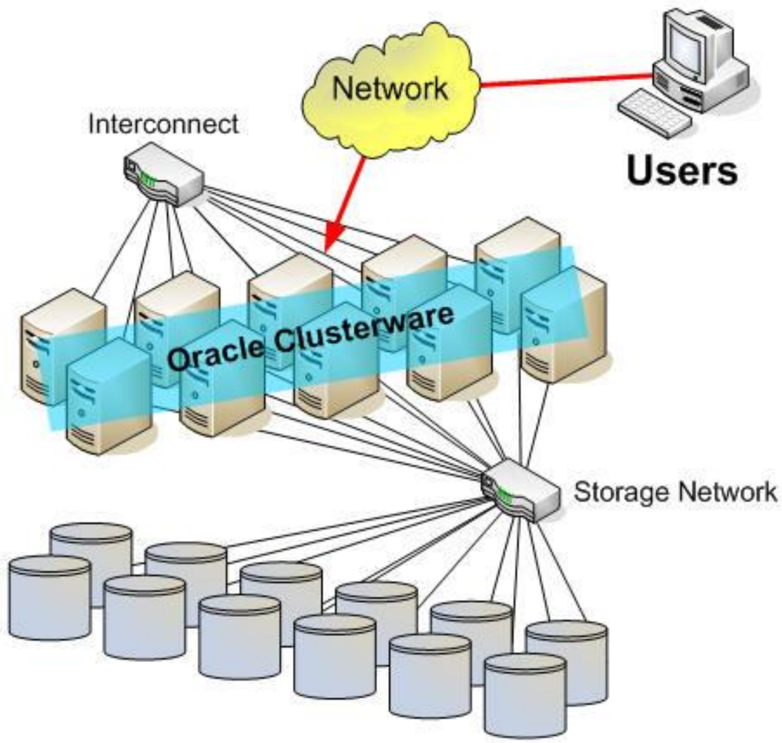
Clusterware

- Clusterware is a software that provides various interfaces and services for a cluster.
- Typically, this includes capabilities that:
  – Allow the cluster to be managed as a whole
  – Protect the integrity of the cluster
  – Maintain a registry of resources across the cluster
  – Deal with changes to the cluster
  – Provide a common view of resources

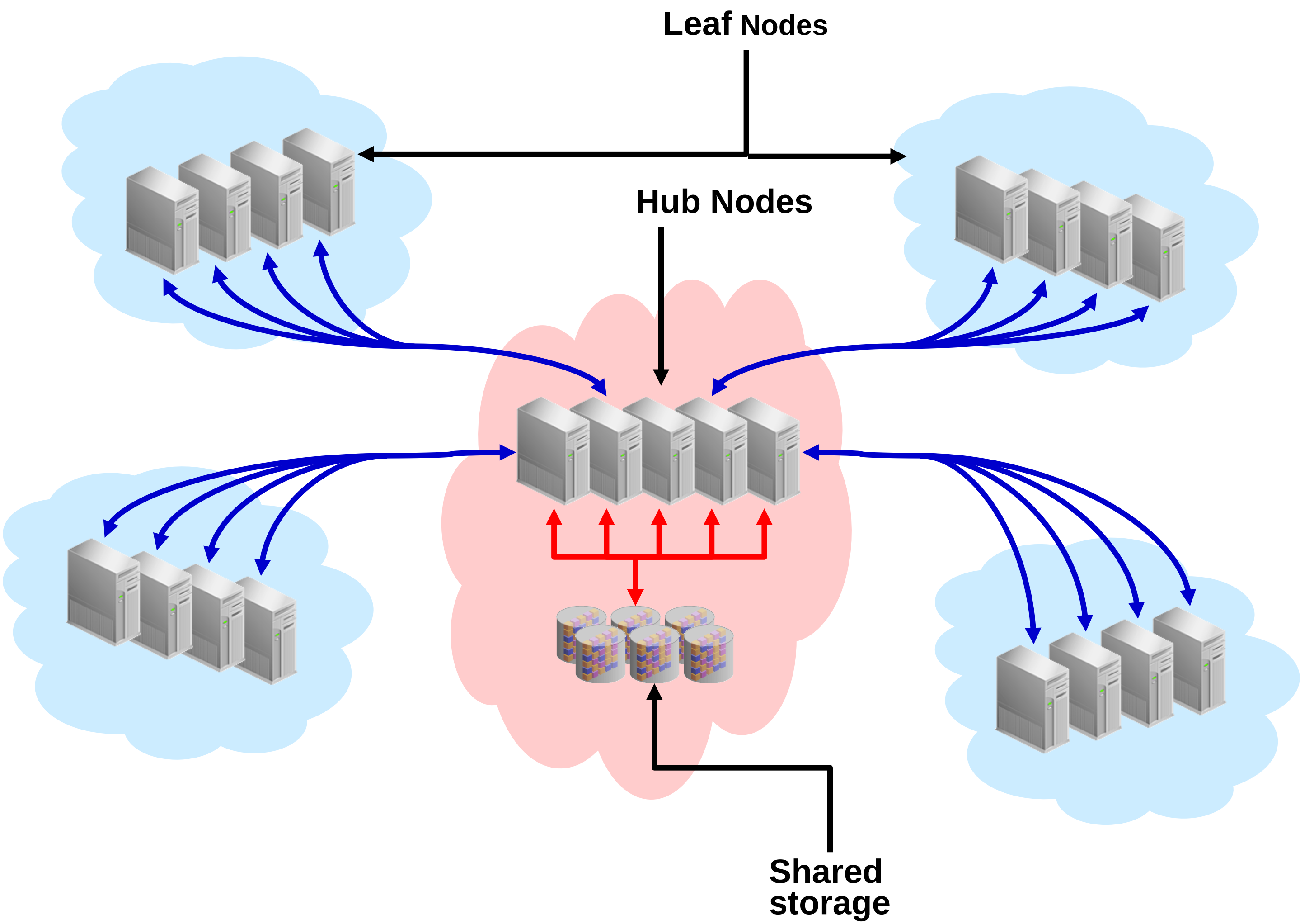
Oracle Clusterware

Oracle Clusterware is:

- A key part of Oracle Grid Infrastructure
- Integrated with Oracle Automatic Storage Management (ASM)
- The basis for Oracle Cloud File System
- A foundation for Oracle Real Application Clusters (RAC)
- A generalized cluster infrastructure for all kinds of applications



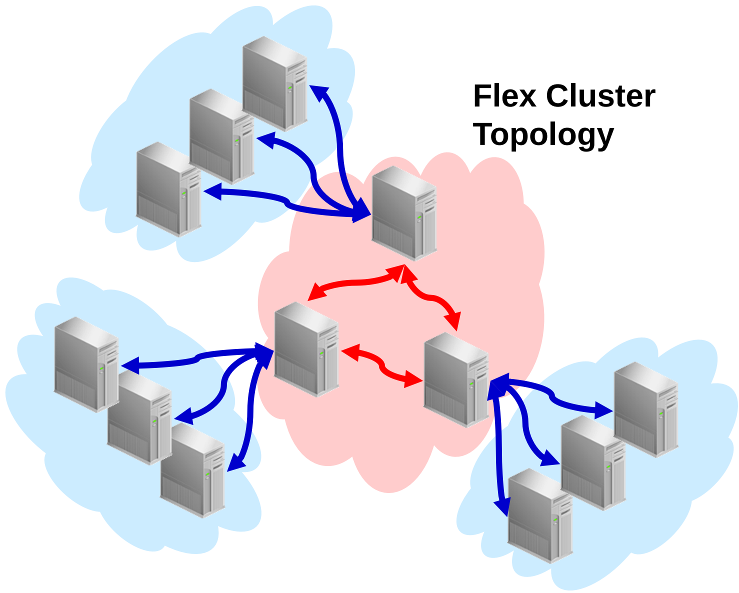
Oracle Flex Clusters

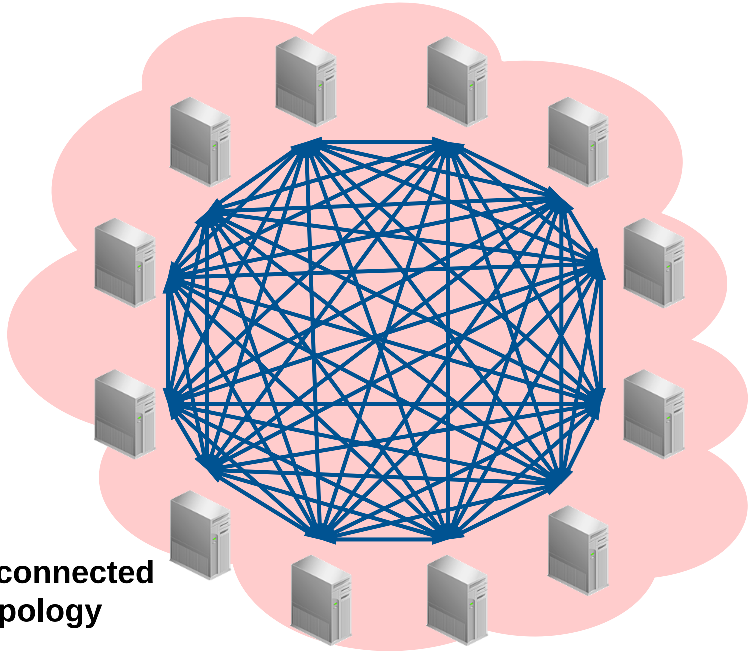


Flex Cluster Scalability

The Flex Cluster hub-and-spoke topology segments the cluster into more manageable groups of nodes.

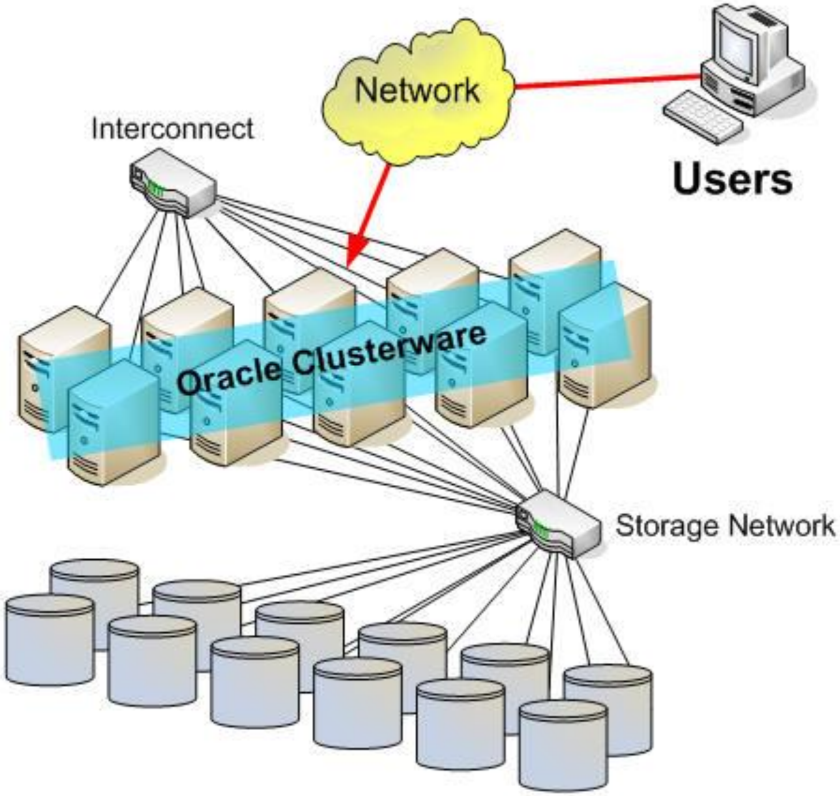
- Only the Hub Nodes require direct access to the OCR and voting disks.
- Fewer interactions are required between nodes.





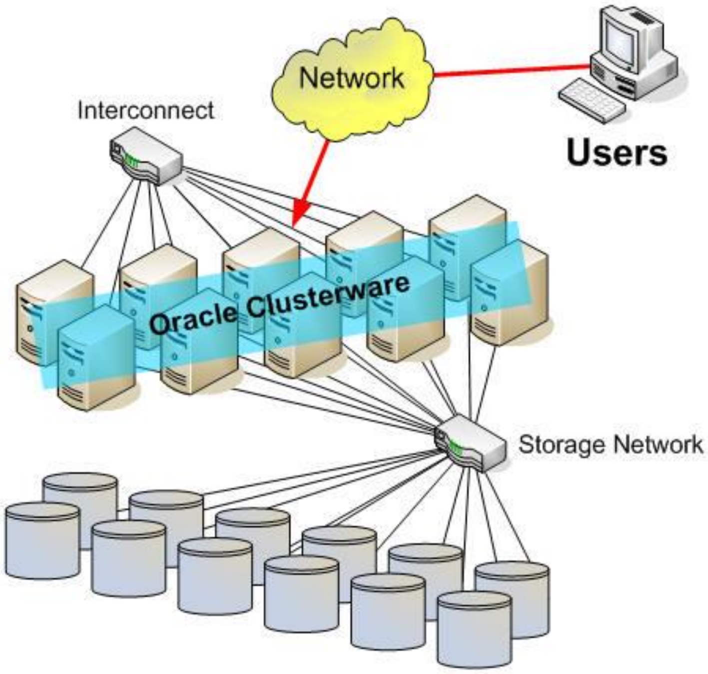
Clusterware Architecture and Cluster Services

- Shared disk cluster architecture supporting application load balancing and failover
- Services include:
  - Cluster management
  - Node monitoring
  - Event services
  - Time synchronization
  - Network management
  - High availability
  - Cluster Interconnect Link Aggregation (HAIP)



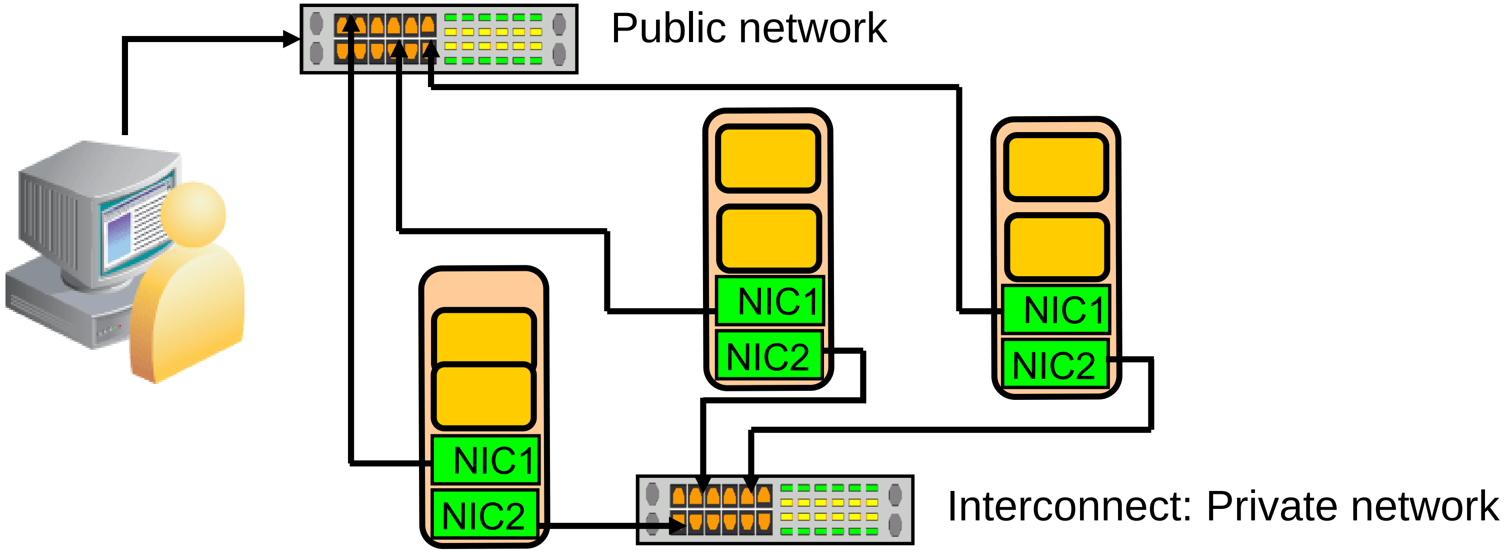
Features of Oracle Clusterware

- Easy installation
- Easy management
- Continuing tight integration with Oracle RAC
- ASM enhancements with benefits for all applications
- No additional clusterware required



Oracle Clusterware Networking

- Each node must have at least two network adapters.
- Each public network adapter must support TCP/IP.
- The interconnect adapter must support:
  - User Datagram Protocol (UDP) or Reliable Data Socket (RDS) for UNIX and Linux for database communication
  - TCP for Windows platforms for database communication
- All platforms use Grid Interprocess Communication (GIPc).



Oracle Clusterware Initialization

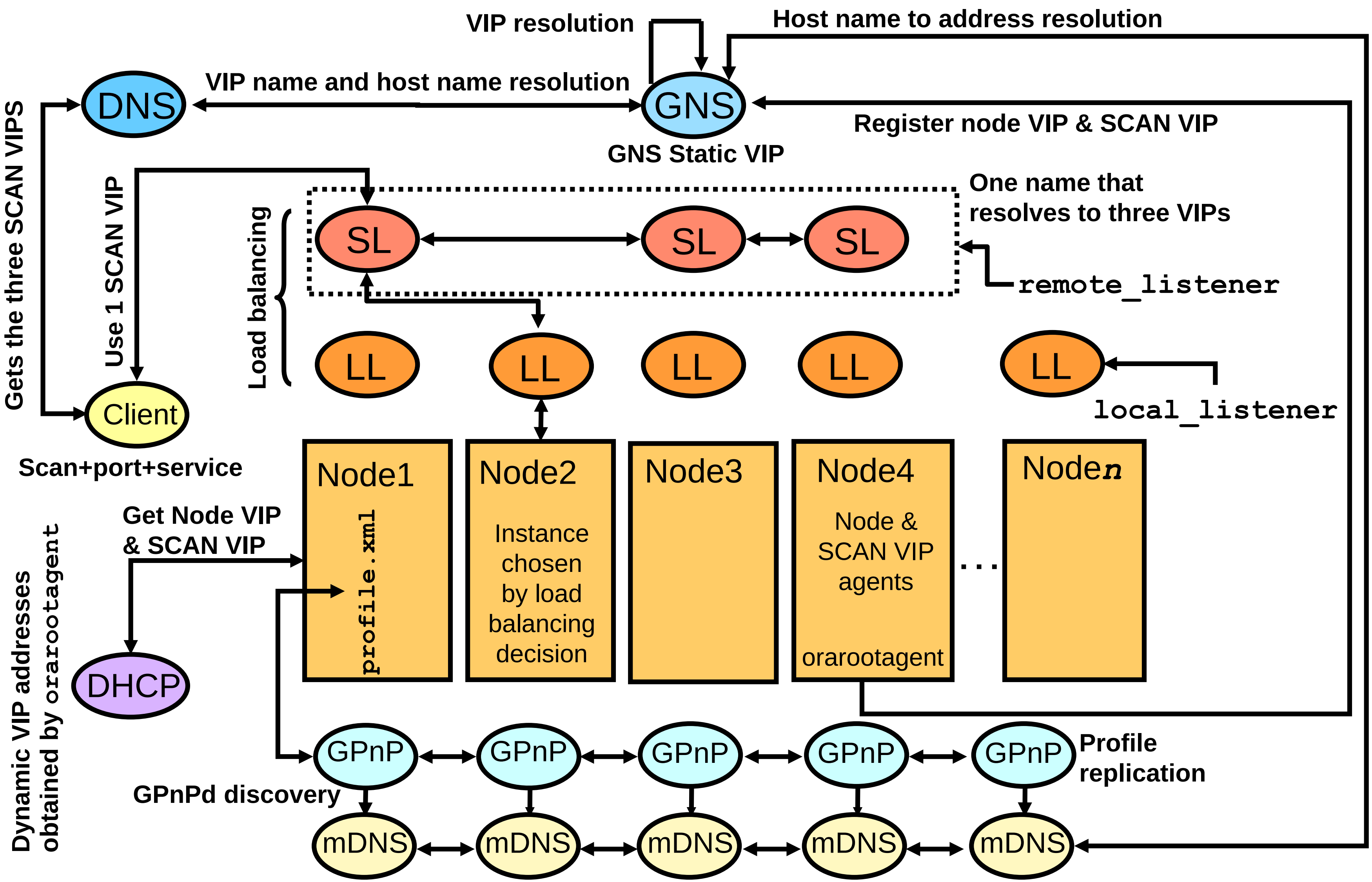
- Oracle Clusterware is started by the OS `init` daemon calling the `/etc/init.d/init.ohasd` startup script.
- On OL5, Oracle Clusterware installation modifies `/etc/inittab` to restart `ohasd` in the event of a crash.

```
# cat /etc/inittab
..
h1:35:respawn:/etc/init.d/init.ohasd run >/dev/null 2>&1 </dev/null
```

- On OL6, Clusterware startup is controlled by Upstart via the `/etc/init/oracle-ohasd.conf` file.

```
# cat /etc/init/oracle-ohasd.conf
# Oracle OHASD startup

start on runlevel [35]
stop  on runlevel [!35]
respawn
exec /etc/init.d/init.ohasd run >/dev/null 2>&1 </dev/null
```

GPnP Architecture: Overview



How GPnP Works: Cluster Node Startup

1. IP addresses are negotiated for public interfaces using DHCP:
   – Node VIPs
   – SCAN VIPs
2. A GPnP agent is started from the nodes Clusterware home.
3. The GPnP agent either gets its profile locally or from one of the peer GPnP agents that responds.
4. Shared storage is configured to match profile requirements.
5. Service startup is specified in the profile, which includes:
   – Grid Naming Service for external names resolution
   – Single-client access name (SCAN) listener

Grid Naming Service (GNS)

- The only static IP address required for the cluster is the GNS virtual IP address.
- The cluster subdomain is defined as a delegated domain.

```
[root@my-dns-server ~]# cat /etc/named.conf
// Default initial "Caching Only" name server configuration
...
# Delegate to gns on cluster01
cluster01.example.com #cluster sub-domain# NS cluster01-gns.example.com
# Let the world know to go to the GNS vip
cluster01-gns.example.com 192.0.2.155 #cluster GNS Address
```

- A request to resolve cluster01-scan.cluster01.example.com would be forwarded to the GNS on 192.0.2.155.
- Each cluster node runs a multicast DNS (mDNS) process.
- You cannot use GNS with another multicast DNS.
  – If you want to use GNS, disable any third-party mDNS daemons on your system.

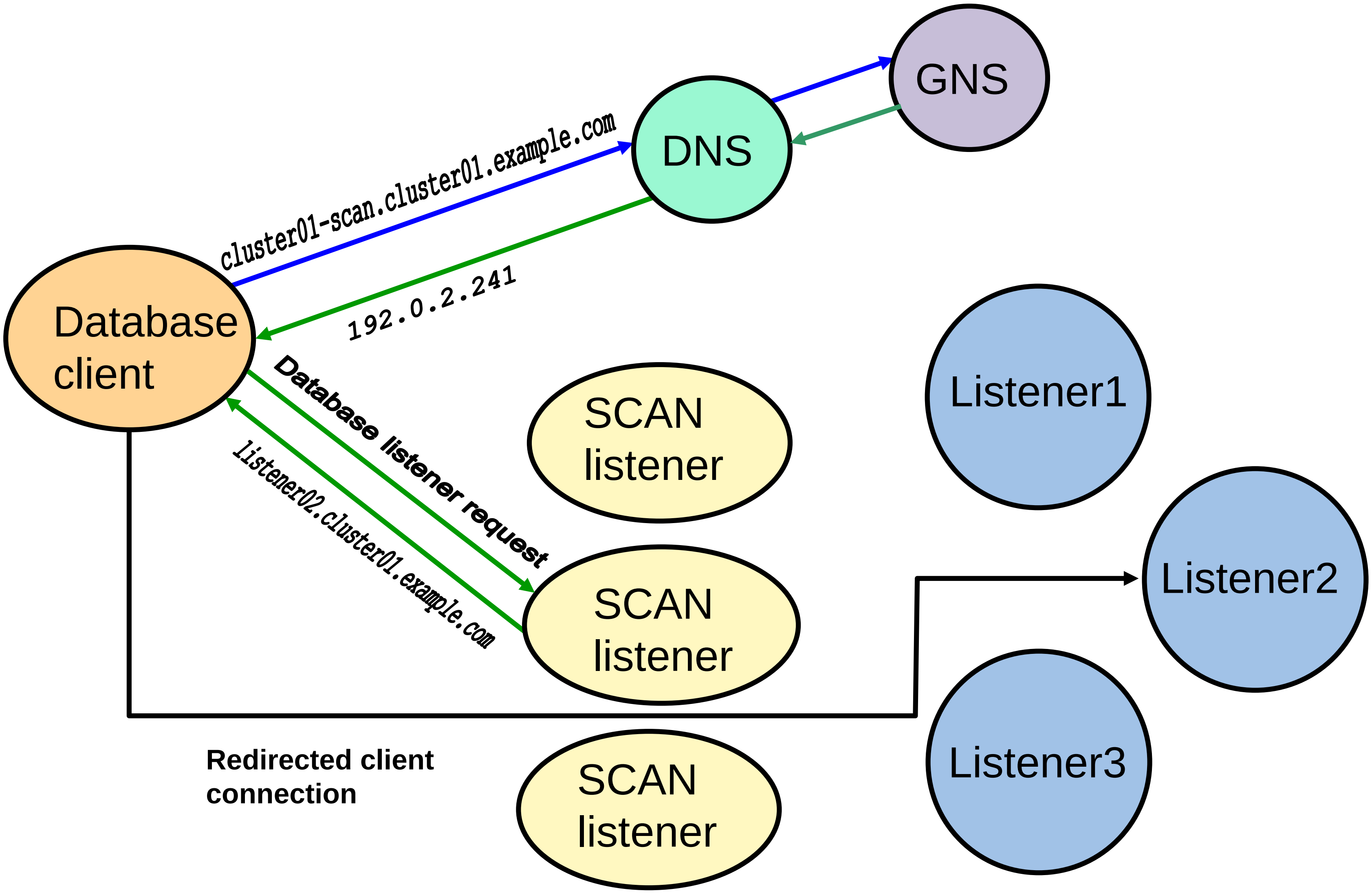
Single-Client Access Name

- The single-client access name (SCAN) is the address used by clients connecting to the cluster.
- The SCAN is a fully qualified host name located in the GNS subdomain registered to three IP addresses.

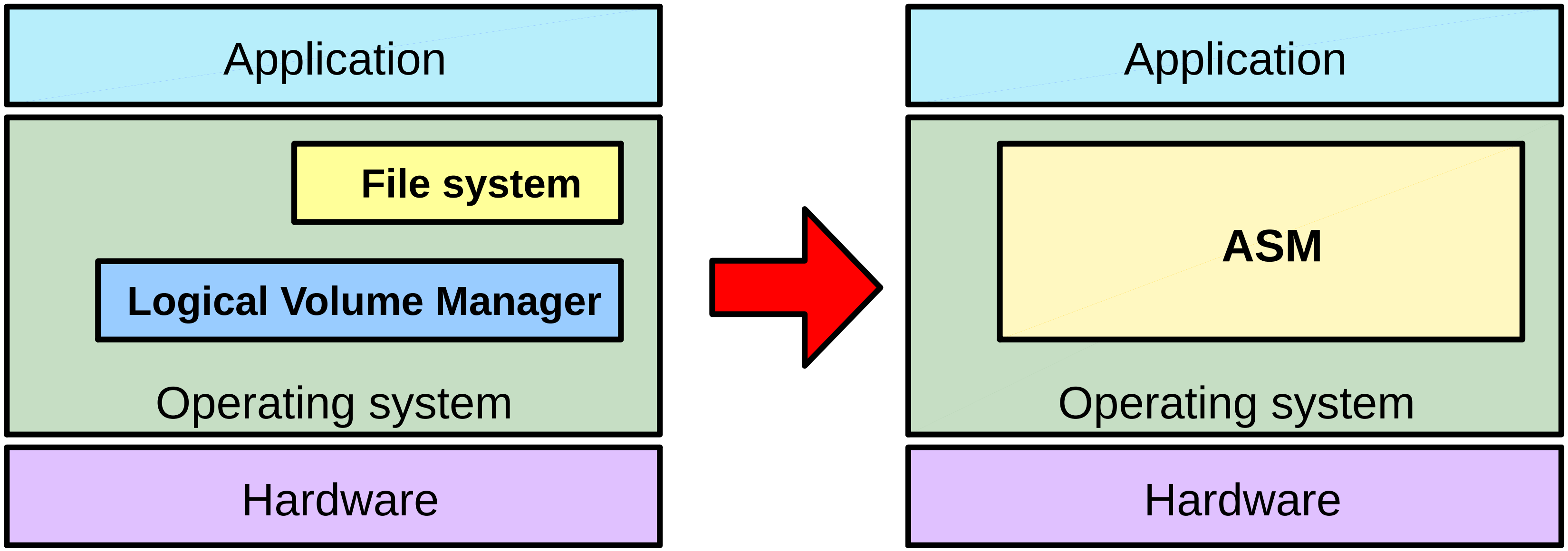
```
$ nslookup cluster01-scan.cluster01.example.com
Server:         192.0.2.1
Address:        192.0.2.1#53

Non-authoritative answer:
Name:   cluster01-scan.cluster01.example.com
Address: 192.0.2.243
Name:   cluster01-scan.cluster01.example.com
Address: 192.0.2.244
Name:   cluster01-scan.cluster01.example.com
Address: 192.0.2.245
```

- The SCAN provides a stable, highly available name for clients to use, independent of the nodes that make up the cluster.

Client Database Connections



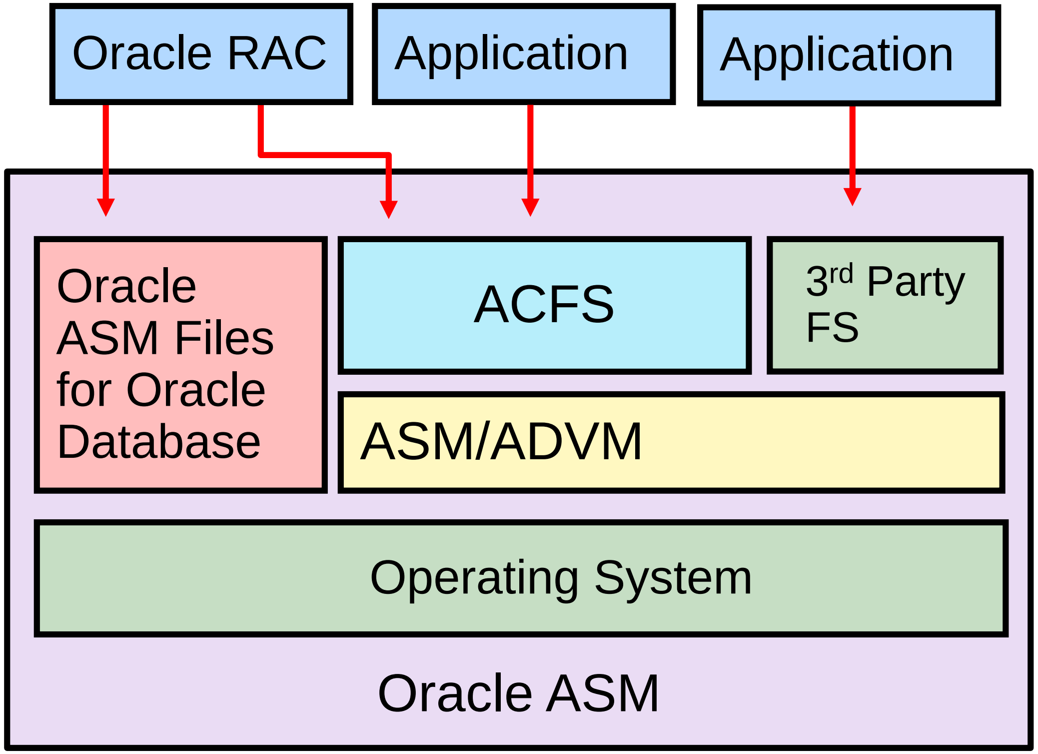
Oracle ASM



Application
File system
Logical Volume Manager
Operating system
Hardware
Application
ASM
Operating system
Hardware

ASM CloudFS and ACFS

- ASM manages Oracle database files.
- ACFS manages other files.
- Spreads data across disks to balance load
- Provides integrated mirroring across disks



Oracle Flex ASM

- Oracle Flex ASM enables an Oracle ASM instance to run on a separate physical server from the database servers.
- Larger clusters of ASM instances can support more clients while reducing the ASM footprint for the overall system.
- With Flex ASM, you can consolidate all the storage requirements into a single set of disk groups.
  - These disk groups are mounted by and managed by a small set of Oracle ASM instances running in a single cluster.
- ASM clients can be configured with direct access to storage or the I/Os can be sent through a pool of I/O servers.

ASM Features and Benefits

- Stripes files rather than logical volumes
- Provides redundancy on a file basis
- Enables online disk reconfiguration and dynamic rebalancing
- Reduces the time significantly to resynchronize a transient failure by tracking changes while the disk is offline
- Provides adjustable rebalancing speed
- Is cluster-aware
- Supports reading from mirrored copy instead of primary copy for extended clusters
- Is automatically installed as part of the Grid Infrastructure

Quiz

Which of the following statements regarding Grid Naming Service is *not* true?

a. GNS is an integral component of Grid Plug and Play.
b. Each node in the cluster runs a multicast DNS (mDNS) process.
c. The GNS virtual IP address must be assigned by DHCP.
d. The cluster subdomain is defined as a delegated domain.

Quiz

Each cluster node's public Ethernet adapter must support UDP or RDS.

a. True
b. False

Summary

In this lesson, you should have learned to:

- Explain the principles and purposes of clusters
- Describe the Oracle Clusterware architecture
- Describe how Grid Plug and Play affects Clusterware

Practice 1: Overview

This practice covers the installation of Oracle Grid Infrastructure.

RAC Databases Overview and Architecture

Objectives

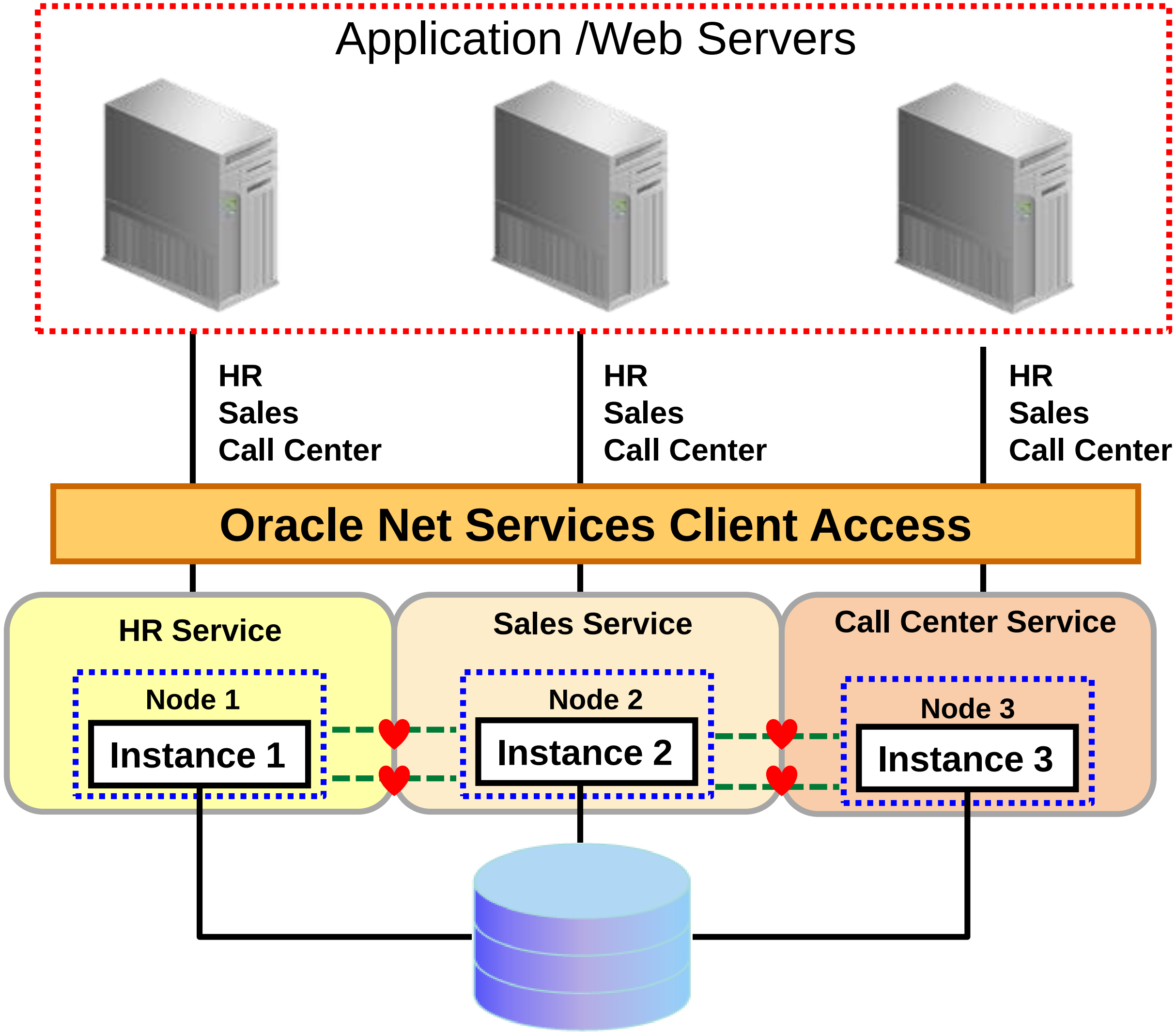
After completing this lesson, you should be able to:

- Describe the benefits of Oracle RAC
- Explain the necessity of global resources
- Describe global cache coordination

Oracle RAC: Overview

- A cluster comprises multiple interconnected servers that appear as one server to end users and applications.
- With Oracle Clusterware, Oracle RAC enables you to cluster an Oracle database.
  - Oracle Clusterware enables nonclustered and RAC databases to use the Oracle high-availability infrastructure.
  - Oracle Clusterware enables you to create a clustered pool of storage to be used by any combination of nonclustered and Oracle RAC databases.
- Noncluster Oracle databases have a one-to-one relationship between the database and the instance.
- Oracle RAC environments have a one-to-many relationship between the database and instances.
  - An Oracle RAC database can have up to 100 instances.

Typical Oracle RAC Architecture



RAC One Node
Single-Instance High Availability

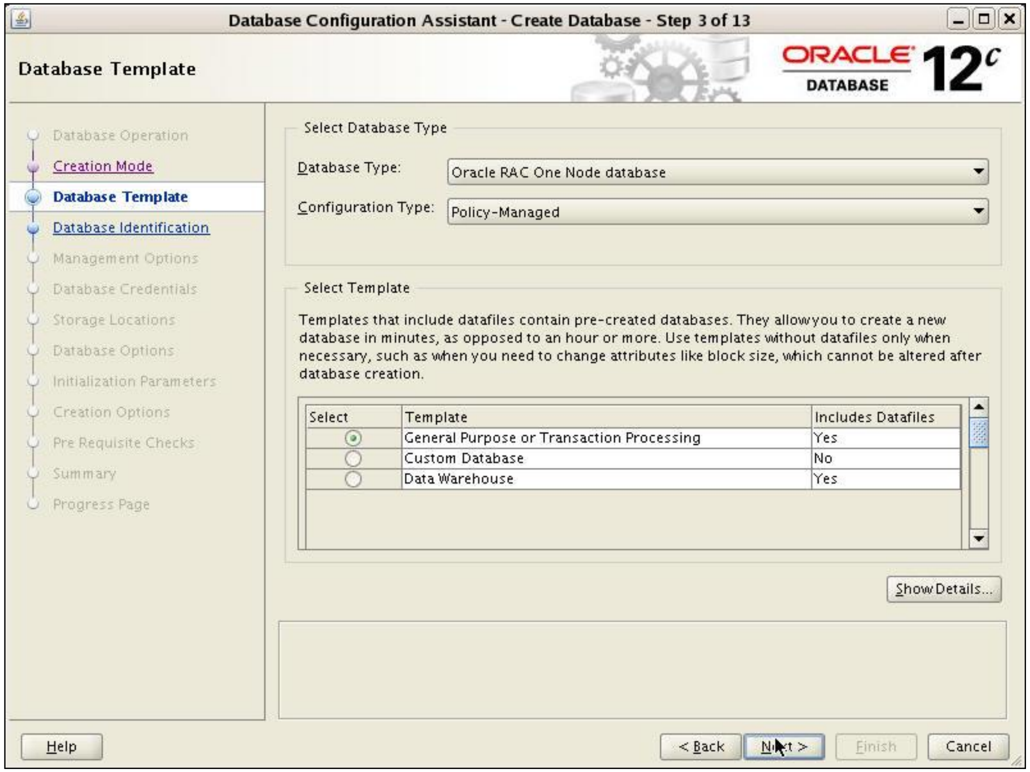
- The Oracle RAC One Node option is a single instance of Oracle RAC running on one node in a cluster.
- This option adds to the flexibility that Oracle offers for consolidation.
- Many databases can be consolidated into a single cluster with minimal overhead while providing:
  - High-availability benefits of failure protection
  - Online rolling patch application
  - Rolling upgrades for the operating system and Oracle Clusterware.
- Oracle RAC One Node is supported on all platforms on which Oracle RAC is certified.

Oracle RAC One Node

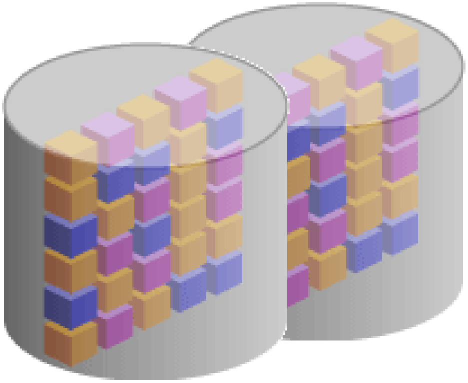
- With online database relocation, a RAC One Node instance can be relocated to another server. This can be useful if:
  - The current server is running short on resources
  - The current server requires maintenance operations, such as operating system patches
- The same technique can be used to relocate RAC One Node instances to high-capacity servers to accommodate changes in workload.
- Single Client Access Name (SCAN) allows clients to connect to the database regardless of where the service is located.
- An Oracle RAC One Node database can be easily scaled up to a full Oracle RAC database if conditions demand it.

Oracle RAC One Node and Oracle Clusterware

- Being closely related to Oracle RAC, Oracle RAC One Node requires Oracle Clusterware.
- You can create an Oracle RAC One Node database with DBCA.



Cluster-Aware Storage Solutions



- RAC databases use a shared-everything architecture and require cluster-aware storage for all database files.
- The Oracle RAC database software manages disk access and is certified for use on a variety of storage architectures.
- Oracle Database supports the following file storage options for Oracle RAC:
  - Oracle Automatic Storage Management
  - Oracle Cluster File System and OCFS2 (Linux)
  - Certified cluster file system or cluster-aware volume manager
  - Certified NFS file servers

Oracle Cluster File System

- It is a shared disk cluster file system for Linux (OCFS2).
- It provides open solution on the operating system side.
- OCFS2 can be downloaded from OTN:
  - http://oss.oracle.com/projects/ocfs2/ (Linux)
- OCFS2 1.6 is the latest version and is now available with the Oracle Linux 5.
- With this release, there are three supported releases of the file system:
  - OCFS2 1.2
  - OCFS2 1.4
  - OCFS2 1.6

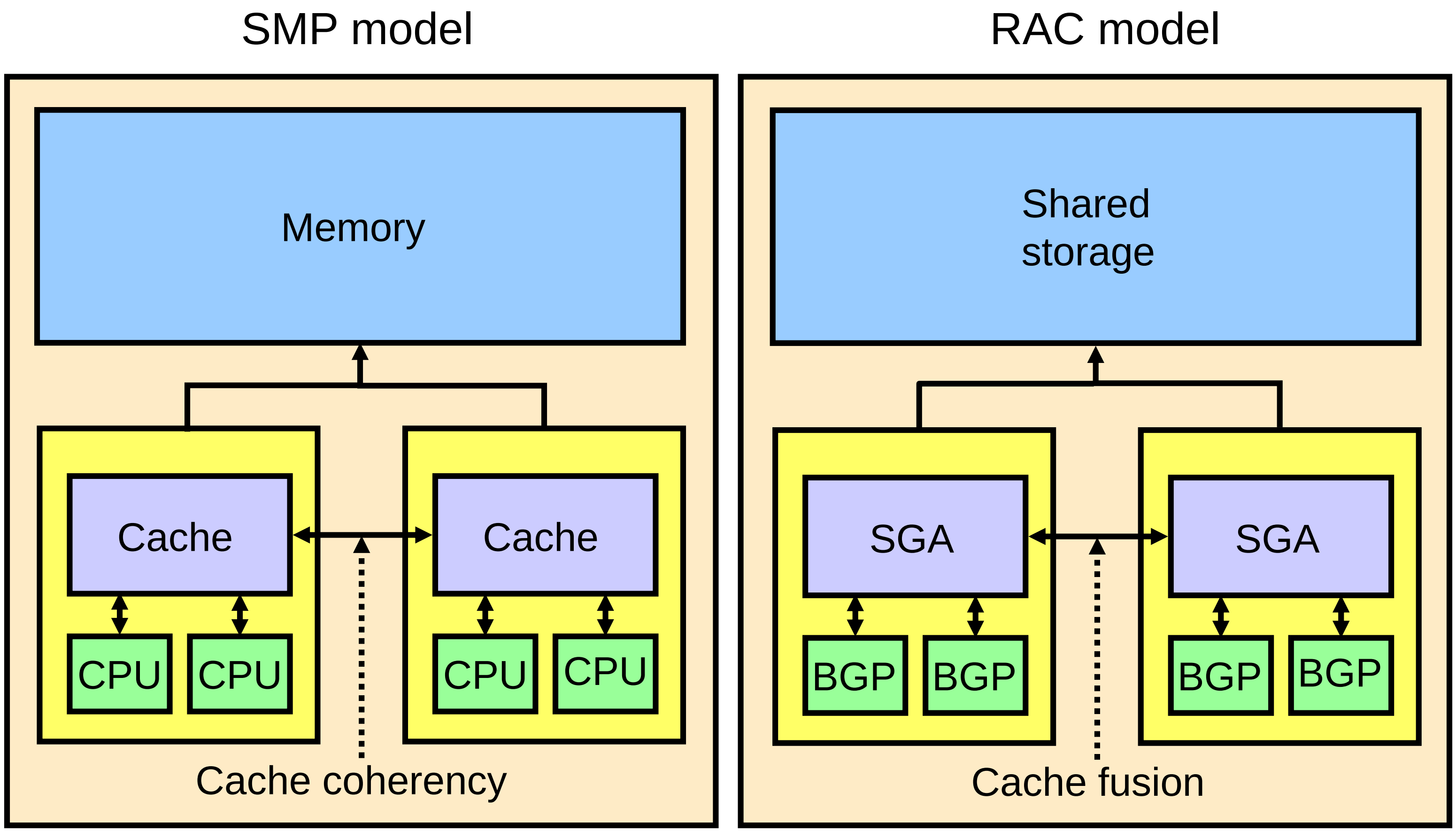
Oracle RAC and Network Connectivity

- All nodes in an RAC environment must connect to at least one Local Area Network, referred to as the public network.
- RAC requires private network connectivity used exclusively for communication between the cluster nodes.
  - The interconnect network is a private network that connects all of the servers in the cluster.
- You must configure UDP for the cluster interconnect on Linux and Unix platforms. Windows clusters use TCP.

Benefits of Using RAC

- High availability: Surviving node and instance failures
- Scalability: Adding more nodes as you need them in the future
- Pay as you grow: Paying for only what you need today
- Key grid computing features:
  - Growth and shrinkage on demand
  - Single-button addition of servers
  - Automatic workload management for services

Clusters and Scalability

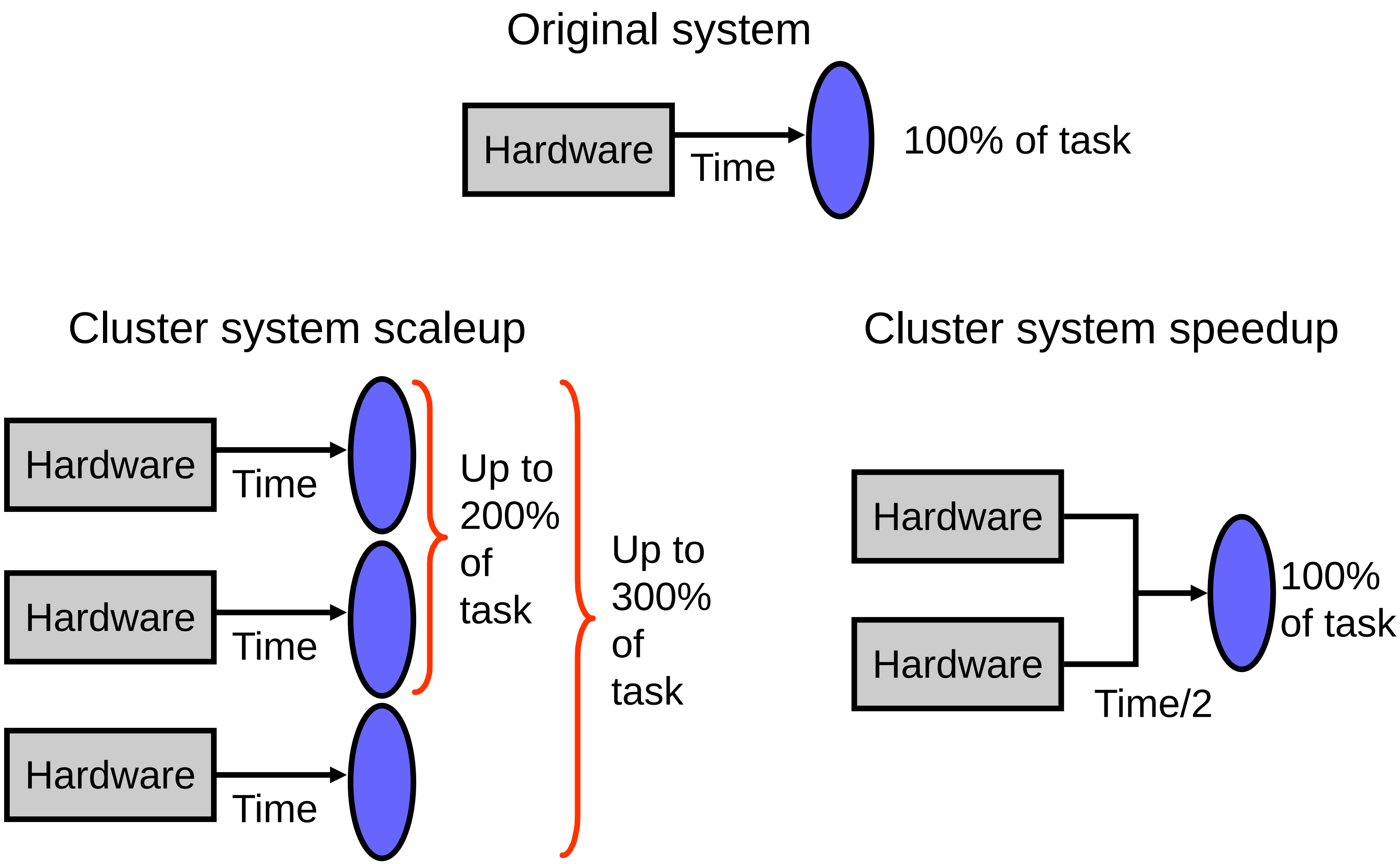


BGP (background process)

Levels of Scalability

- Hardware: Disk input/output (I/O)
- Internode communication: High bandwidth and low latency
- Operating system: Number of CPUs
- Database management system: Synchronization
- Application: Design

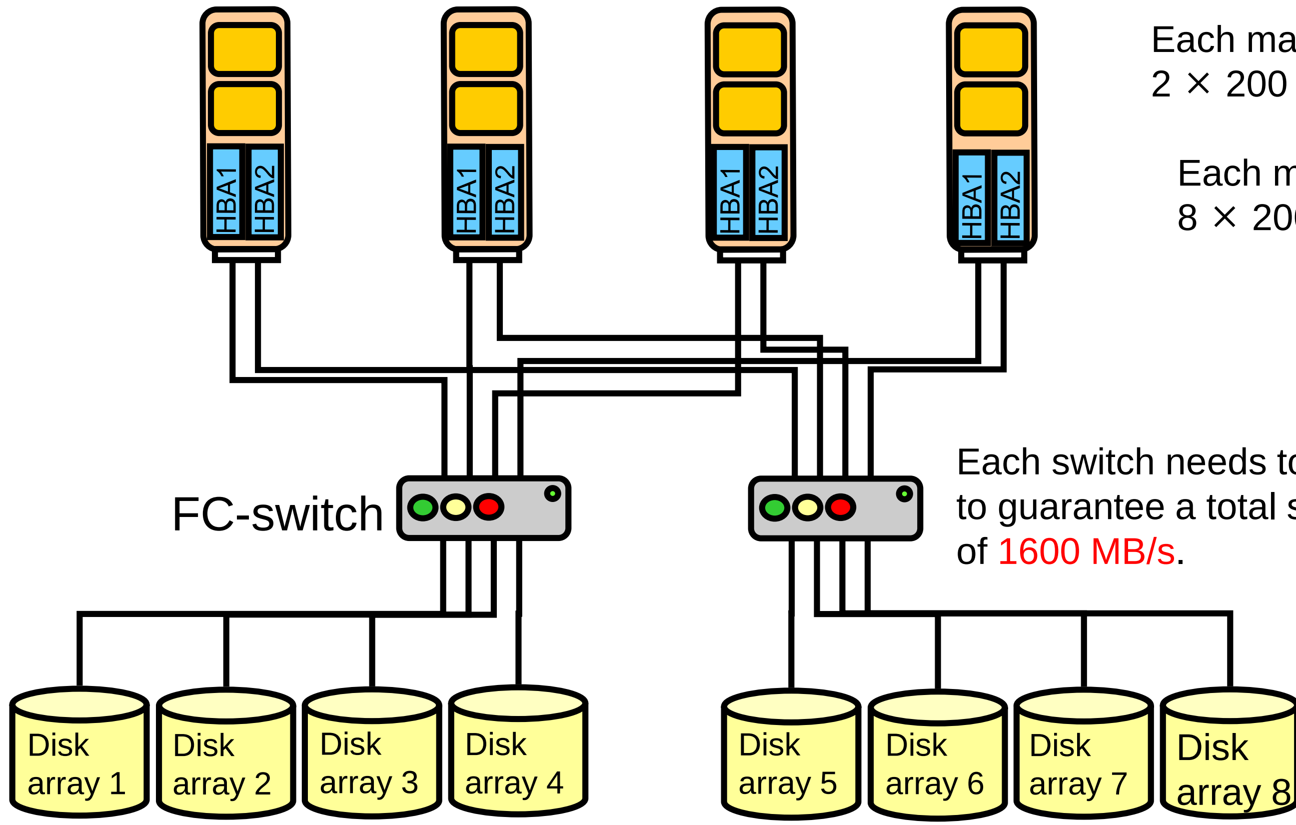
Scaleup and Speedup



Speedup/Scaleup and Workloads

Workload	Speedup	Scaleup
OLTP and Internet	No	Yes
DSS with parallel query	Yes	Yes
Batch (mixed)	Possible	Yes

I/O Throughput Balanced: Example



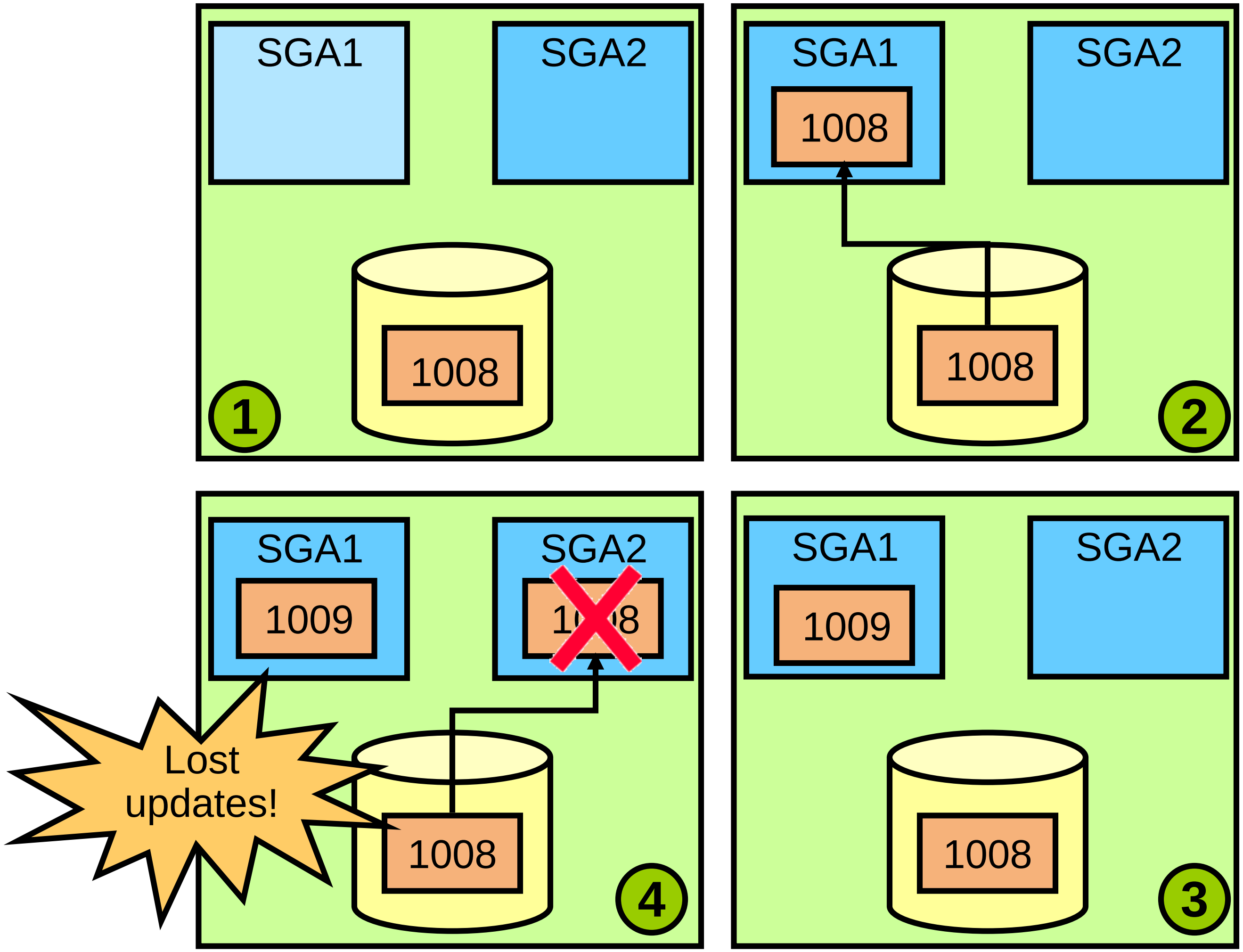
Each machine has 2 CPUs:
2 × 200 MB/s × 4 = 1600 MB/s

Each machine has 2 HBAs:
8 × 200 MB/s = 1600 MB/s

Each switch needs to support 800 MB/s to guarantee a total system throughput of 1600 MB/s.

Each disk array has one 2 Gb controller:
8 × 200 MB/s = 1600 MB/s

Necessity of Global Resources



Additional Memory Requirement for RAC

- Heuristics for scalability cases:
  - 15% more shared pool
  - 10% more buffer cache
- Smaller buffer cache per instance in the case of single-instance workload distributed across multiple instances
- Current values:

```
SELECT resource_name,
       current_utilization,max_utilization
FROM   v$resource_limit
WHERE  resource_name like 'g%s_%';
```

```
SELECT * FROM v$sgastat
WHERE name like 'g_s%' or name like 'KCL%';
```

Parallel Execution with RAC

- In a RAC environment, a SQL statement executed in parallel can run across all of the nodes in the cluster.
- For effective inter-node parallel execution, the interconnect must be size appropriately.
  - Inter-node parallel execution may increase interconnect traffic.
  - If the interconnect bandwidth is less than the disk subsystem, parallel execution should limited to a smaller set of nodes.
- The `PARALLEL_FORCE_LOCAL` parameter is used to limit inter-node parallel execution.
  - When set to `TRUE`, parallel server processes can execute only on the same node where the SQL statement was started.
- Services can be used to limit the number of instances that participate in a parallel SQL operation.

Summary

In this lesson, you should have learned how to:

- Describe the benefits of Oracle RAC
- Explain the necessity of global resources
- Describe global cache coordination

Installing and Configuring Oracle RAC

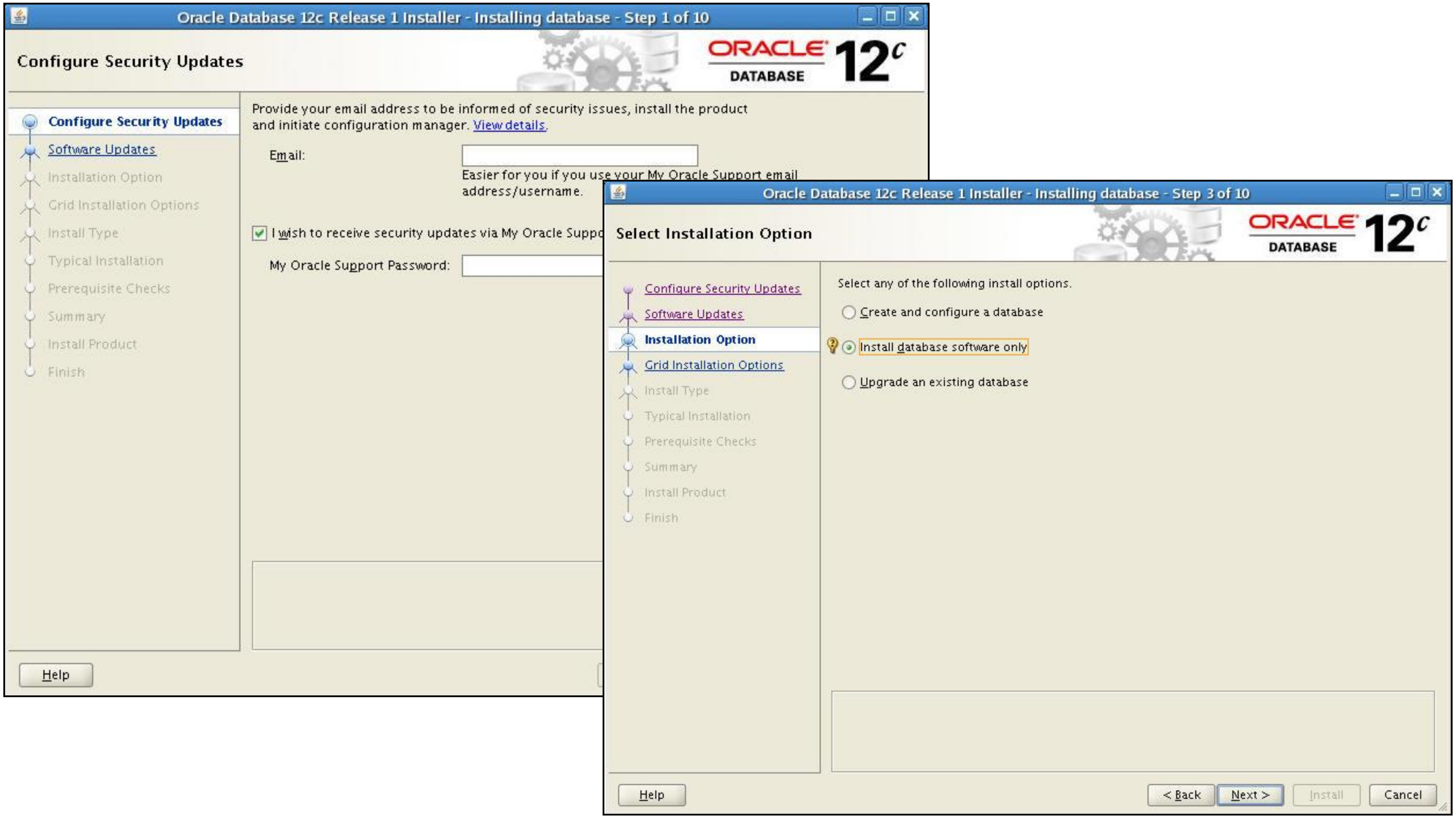
Objectives

After completing this lesson, you should be able to:

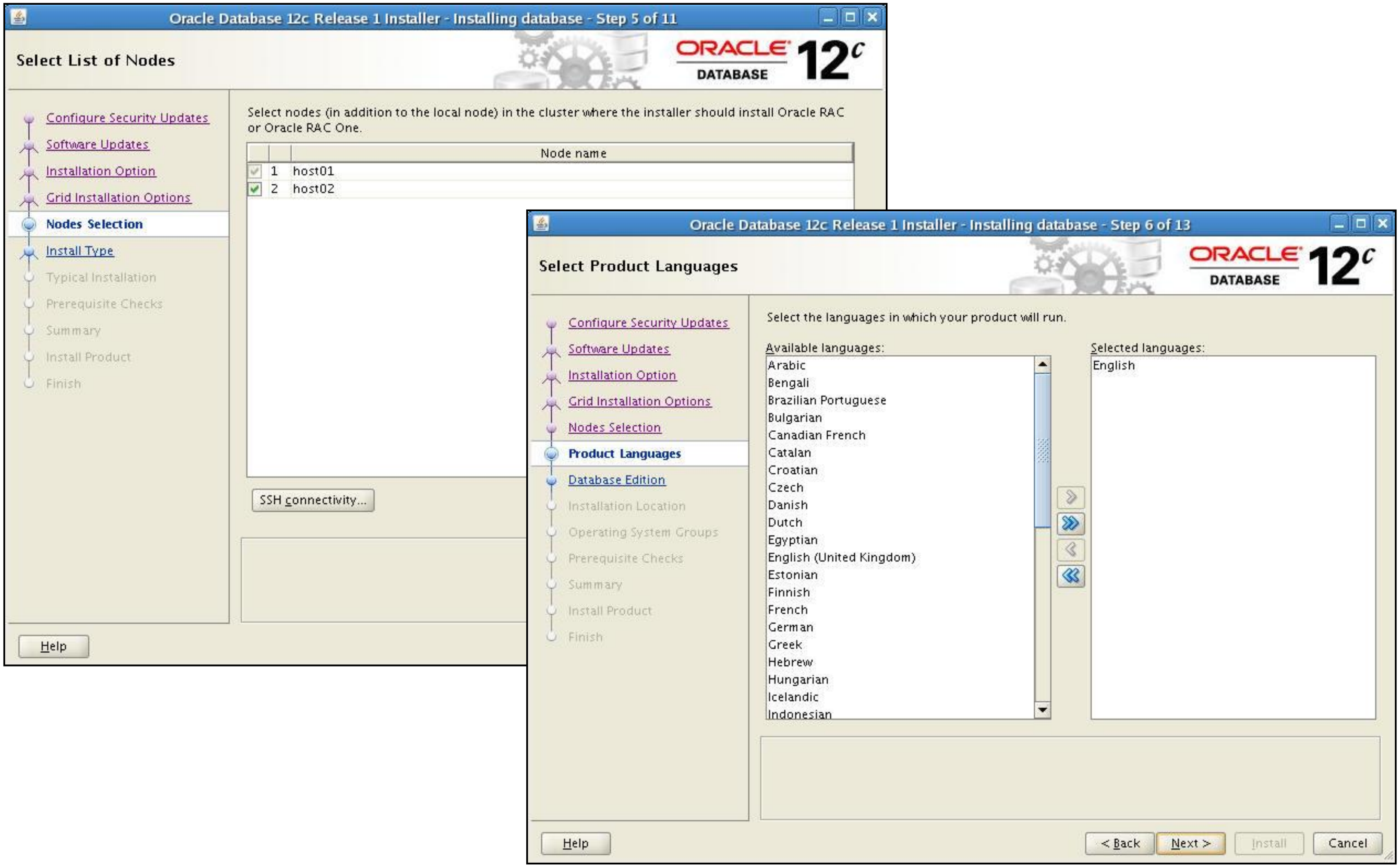
- Install the Oracle database software
- Create a cluster database
- Perform post-database-creation tasks
- Convert a single-instance Oracle database to RAC

Installing the Oracle Database Software

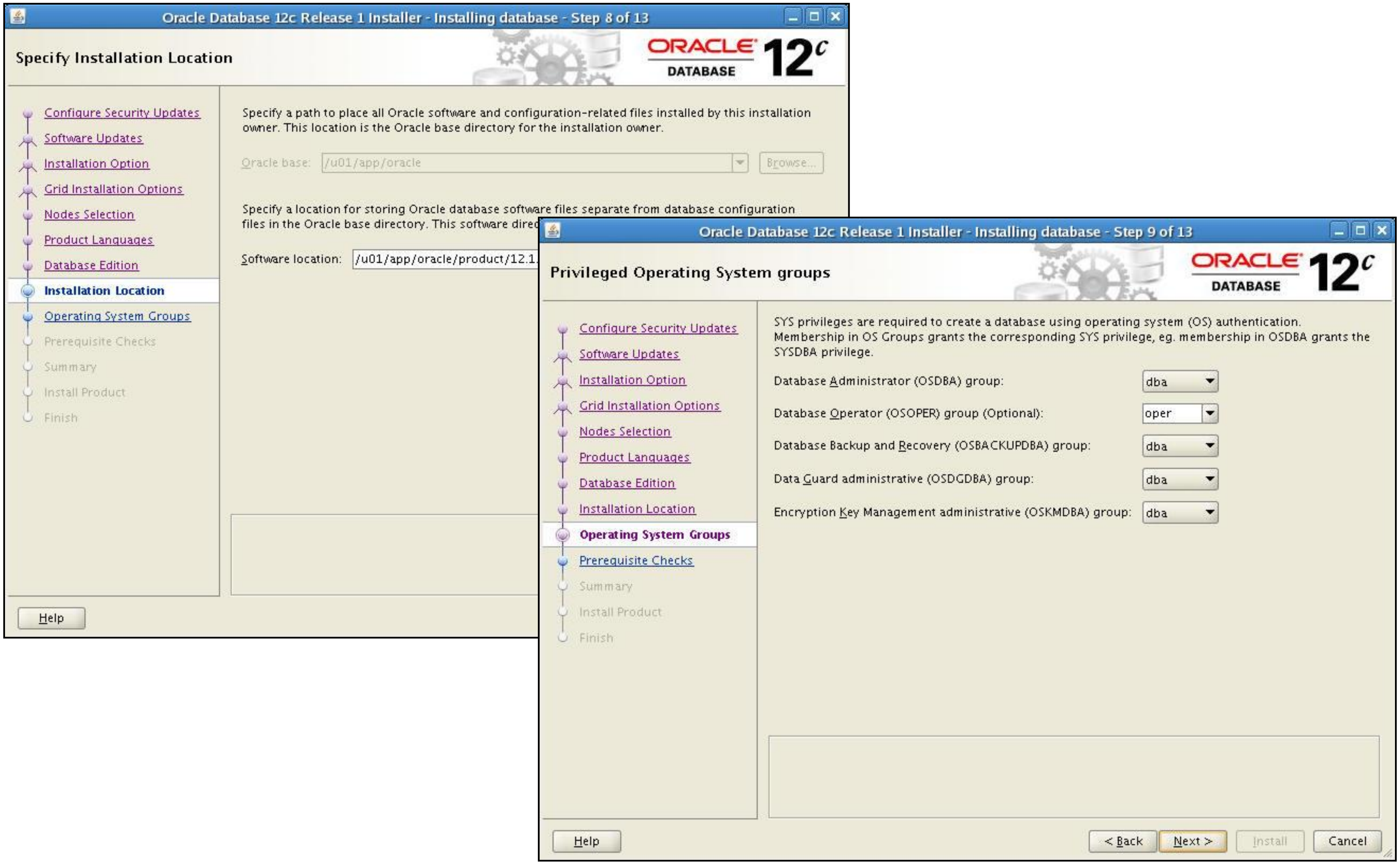
```
$ /stage/database/Disk1/runInstaller
```



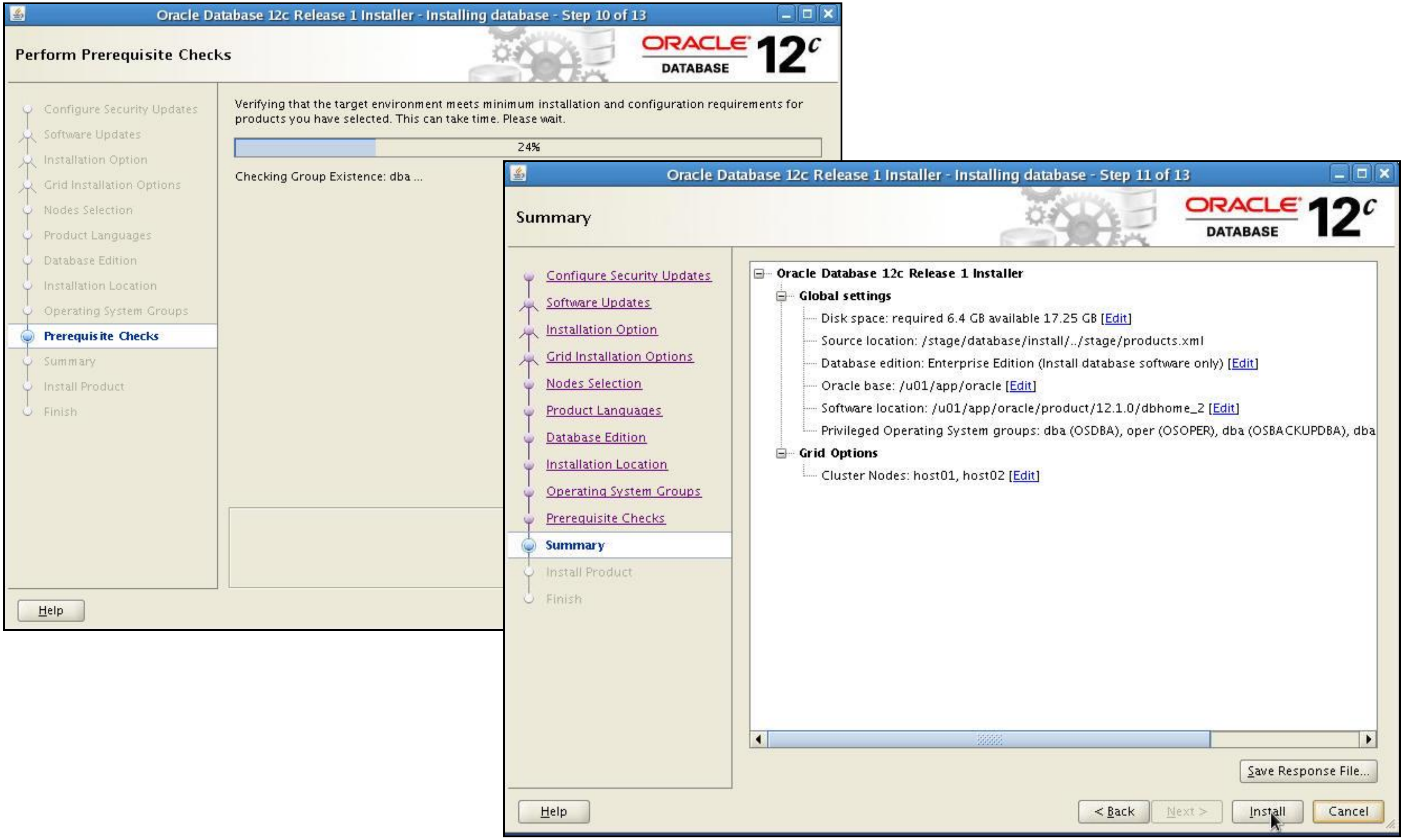
Installing the Oracle Database Software



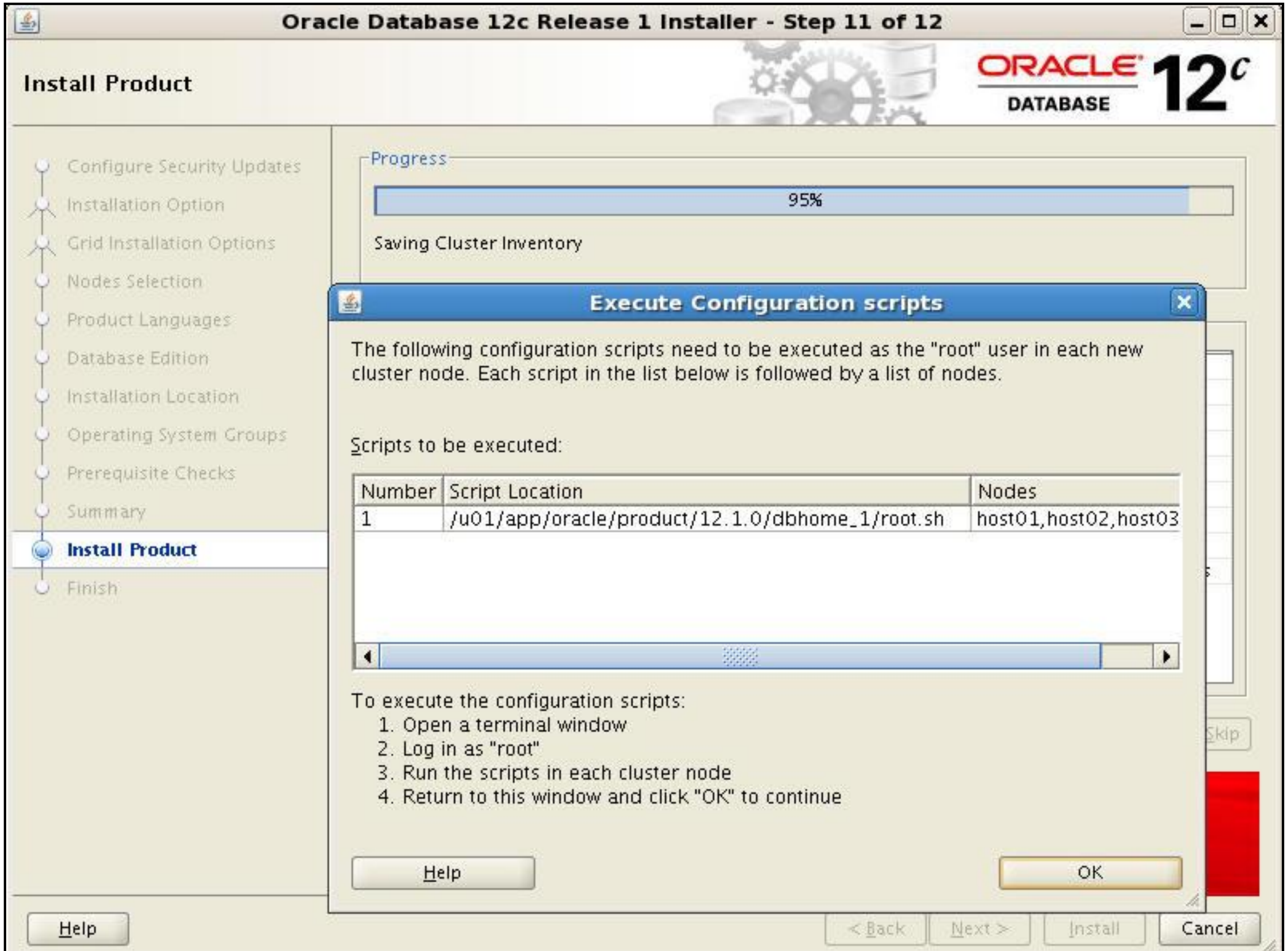
Installing the Oracle Database Software



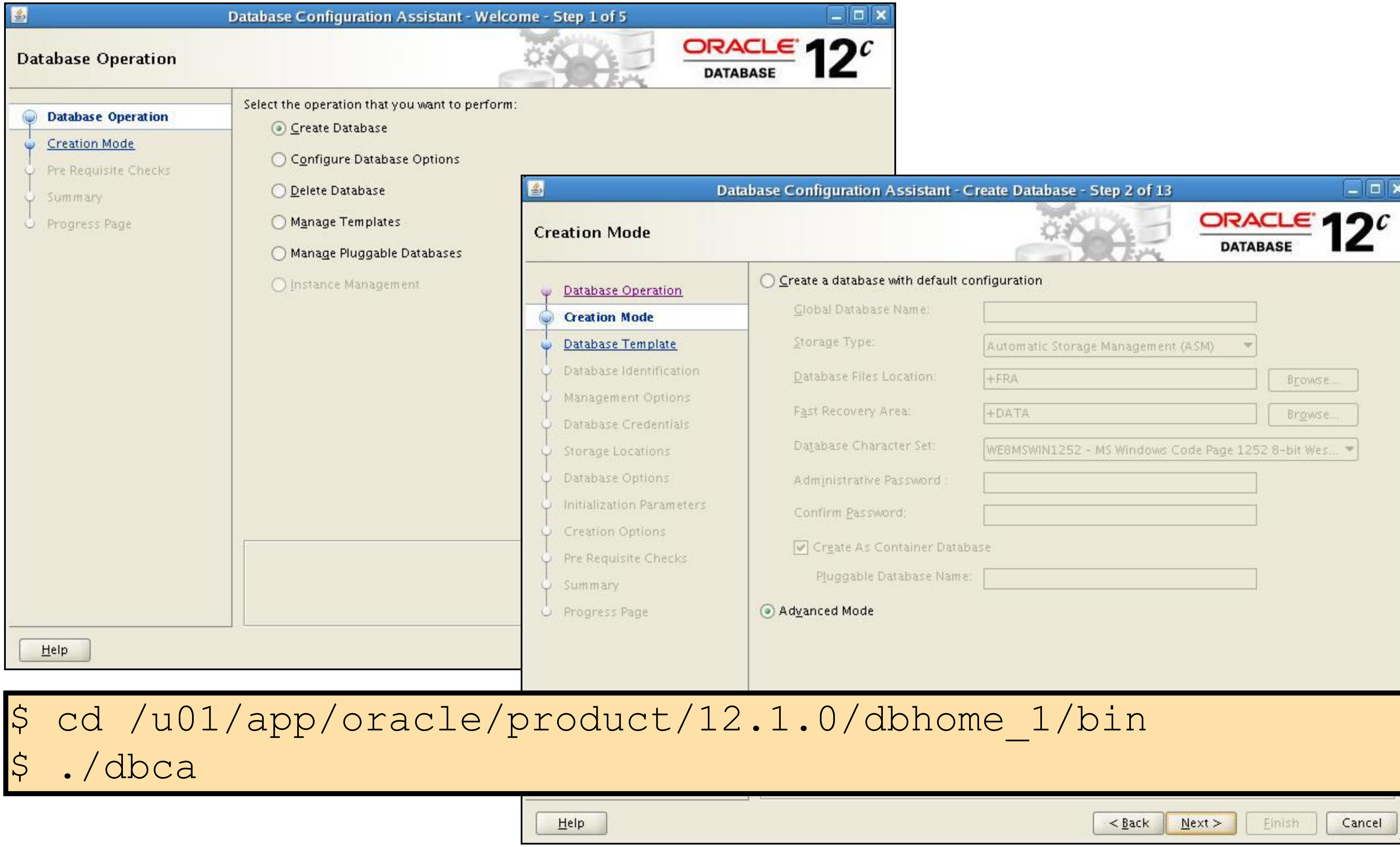
Installing the Oracle Database Software



Installing the Oracle Database Software

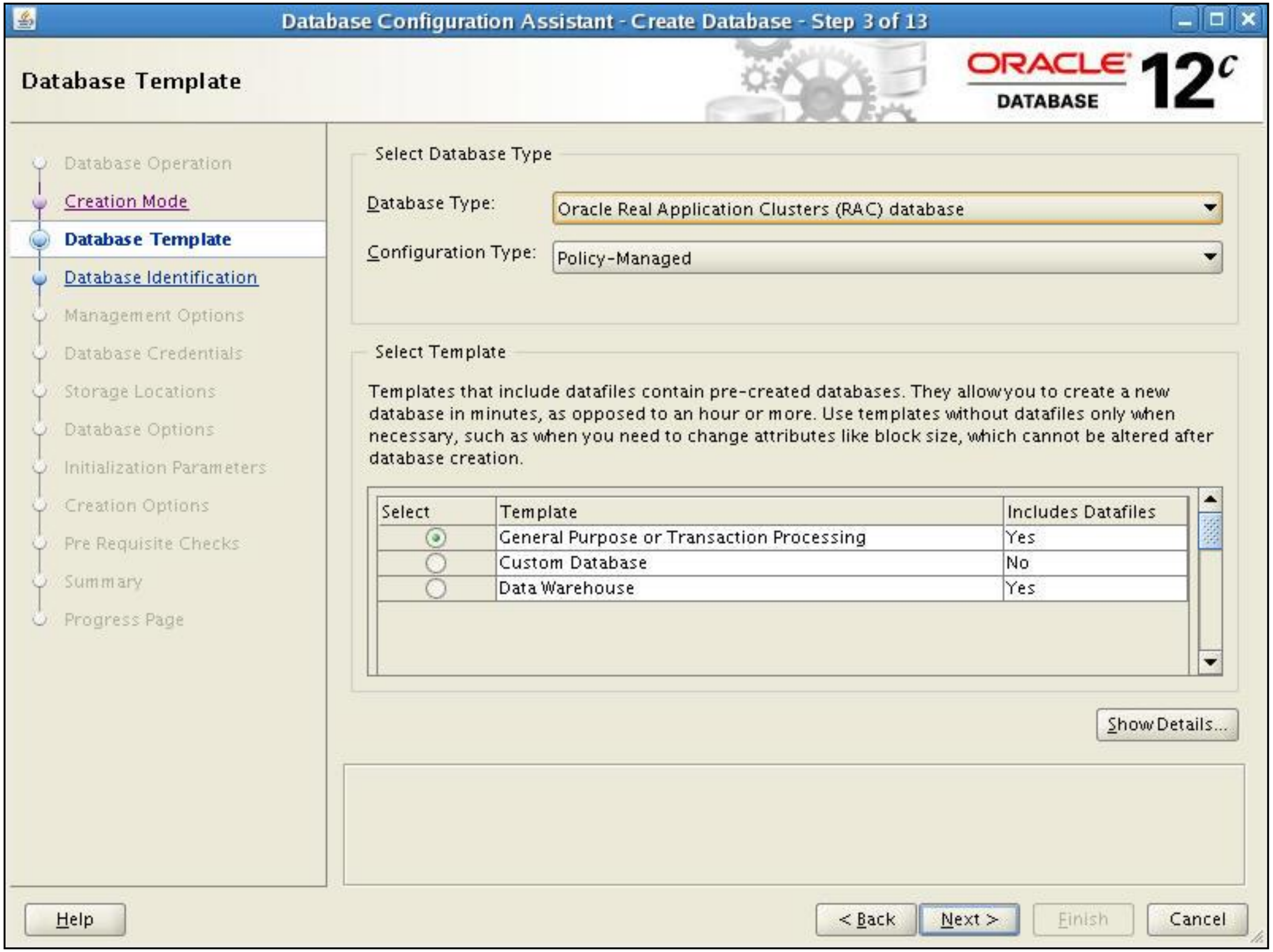


Creating the Cluster Database

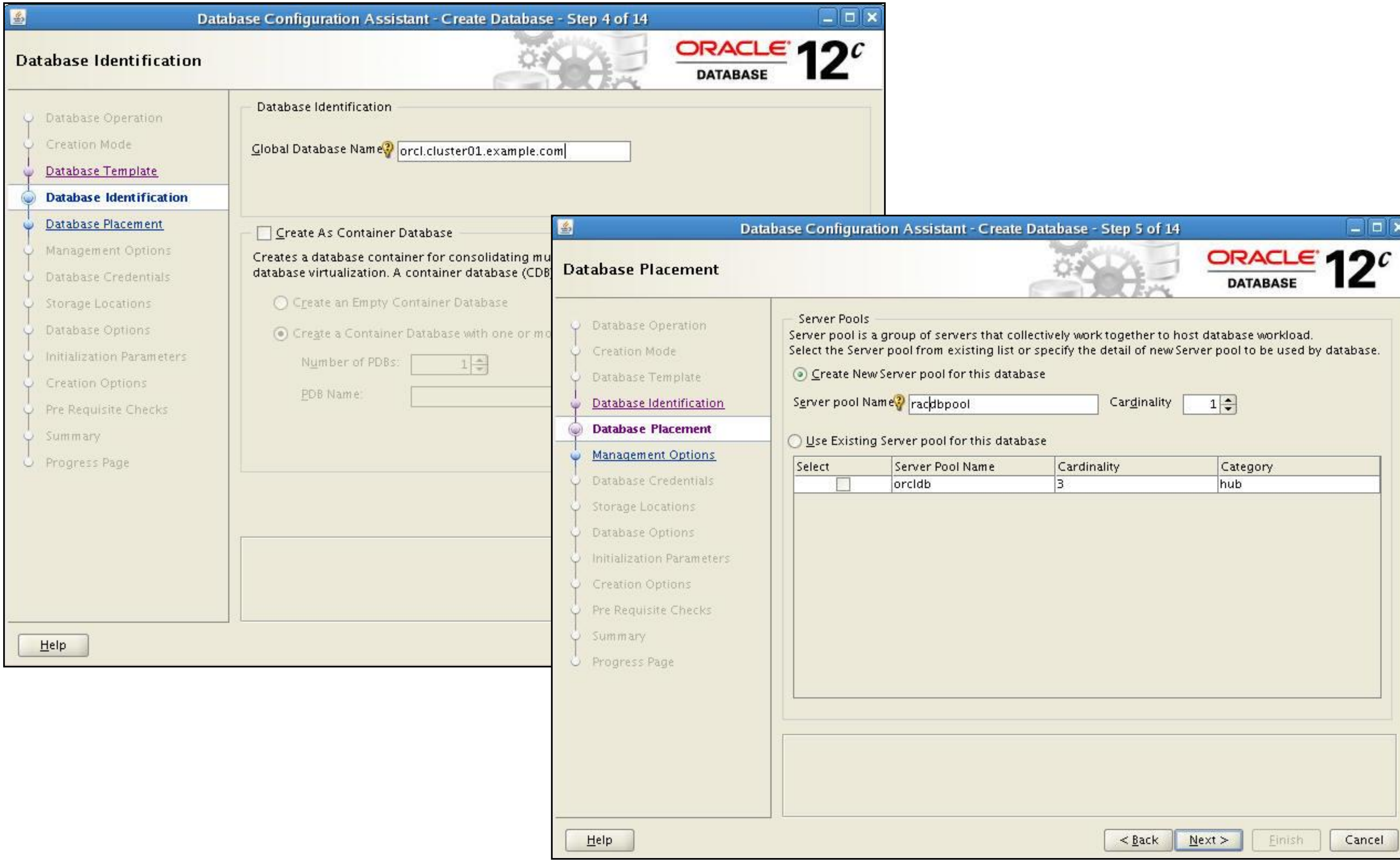



```
$ cd /u01/app/oracle/product/12.1.0/dbhome_1/bin
$ ./dbca
```

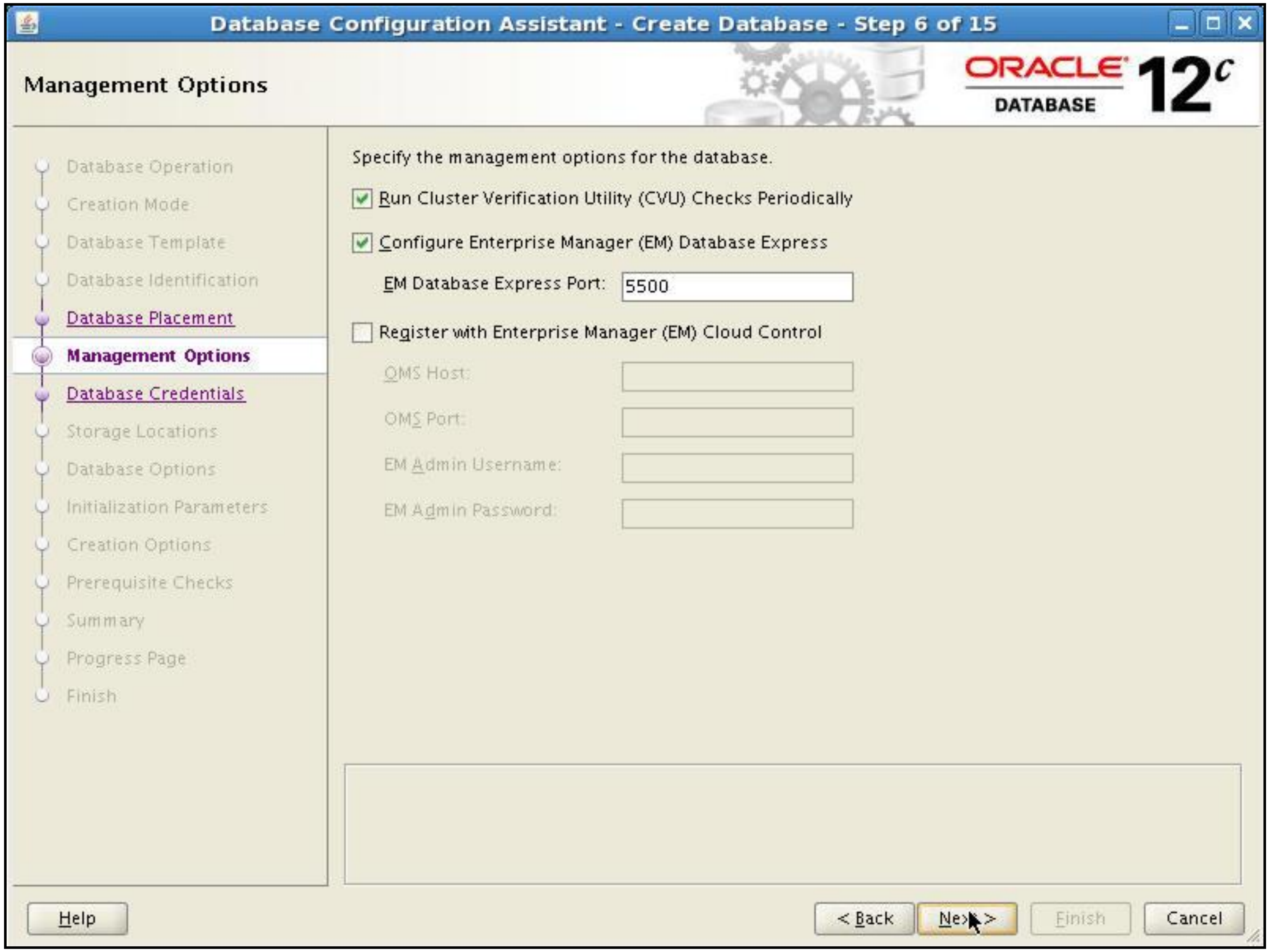
Database Type Selection



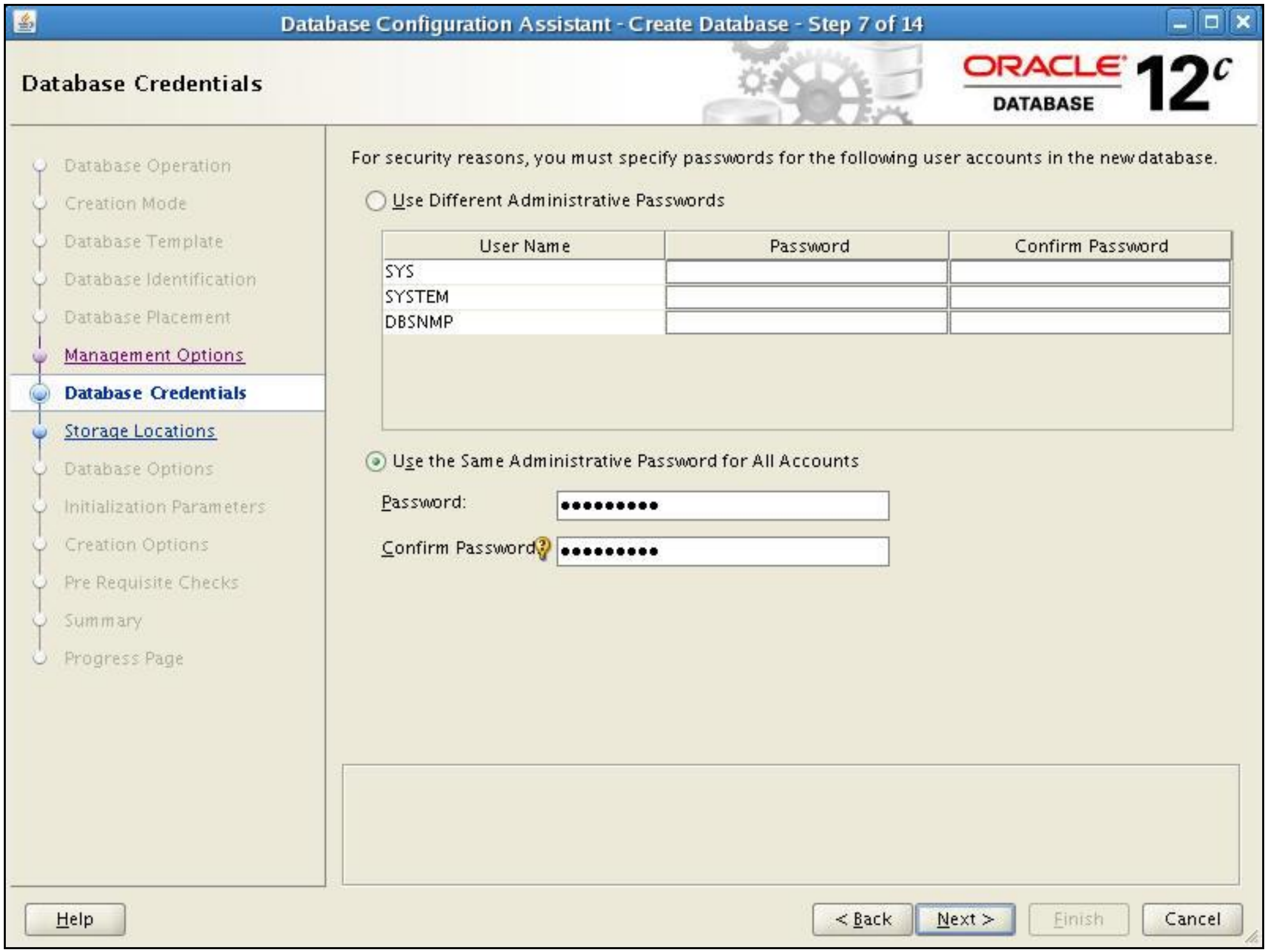
Database Identification



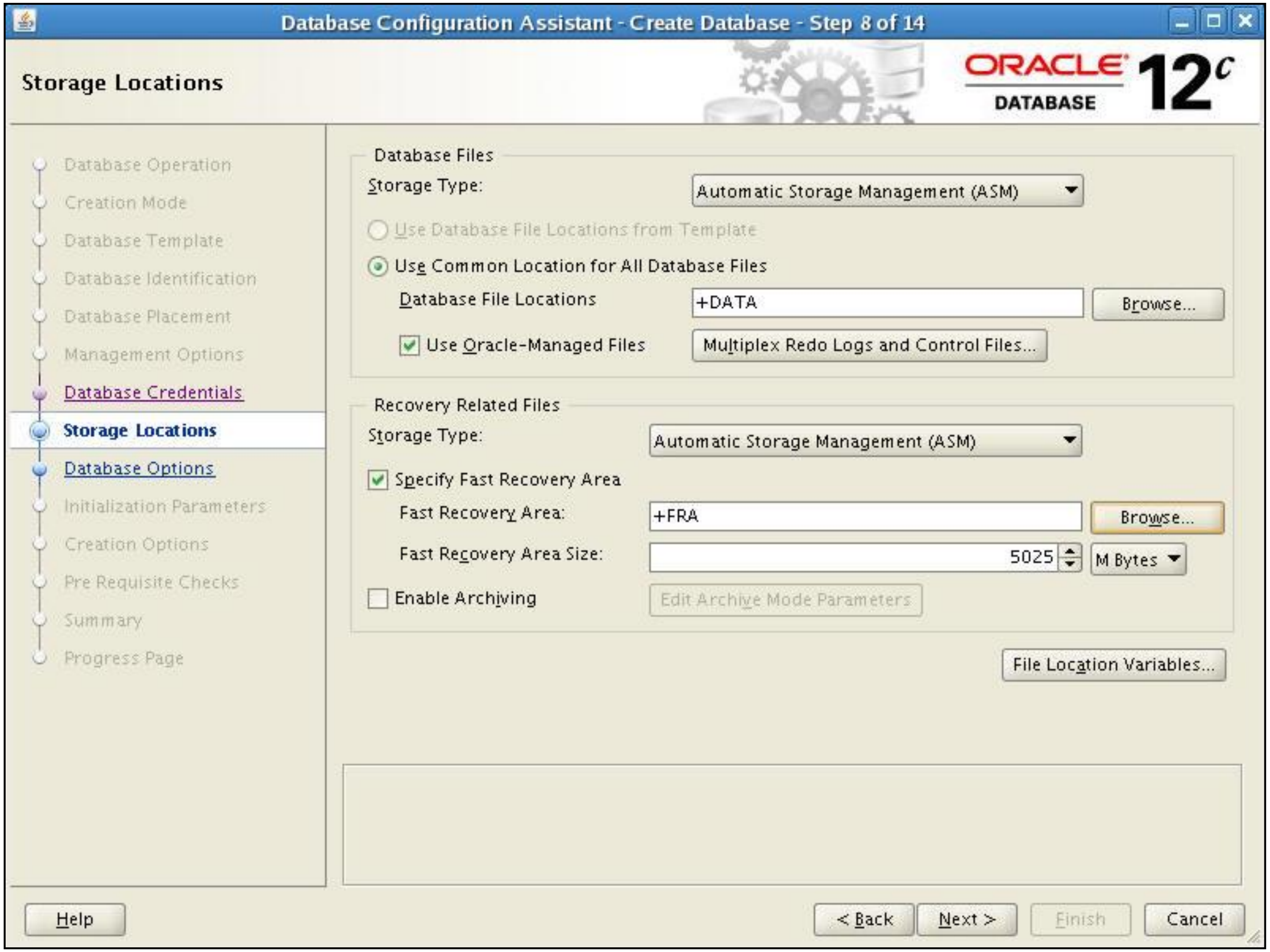
Cluster Database Management Options



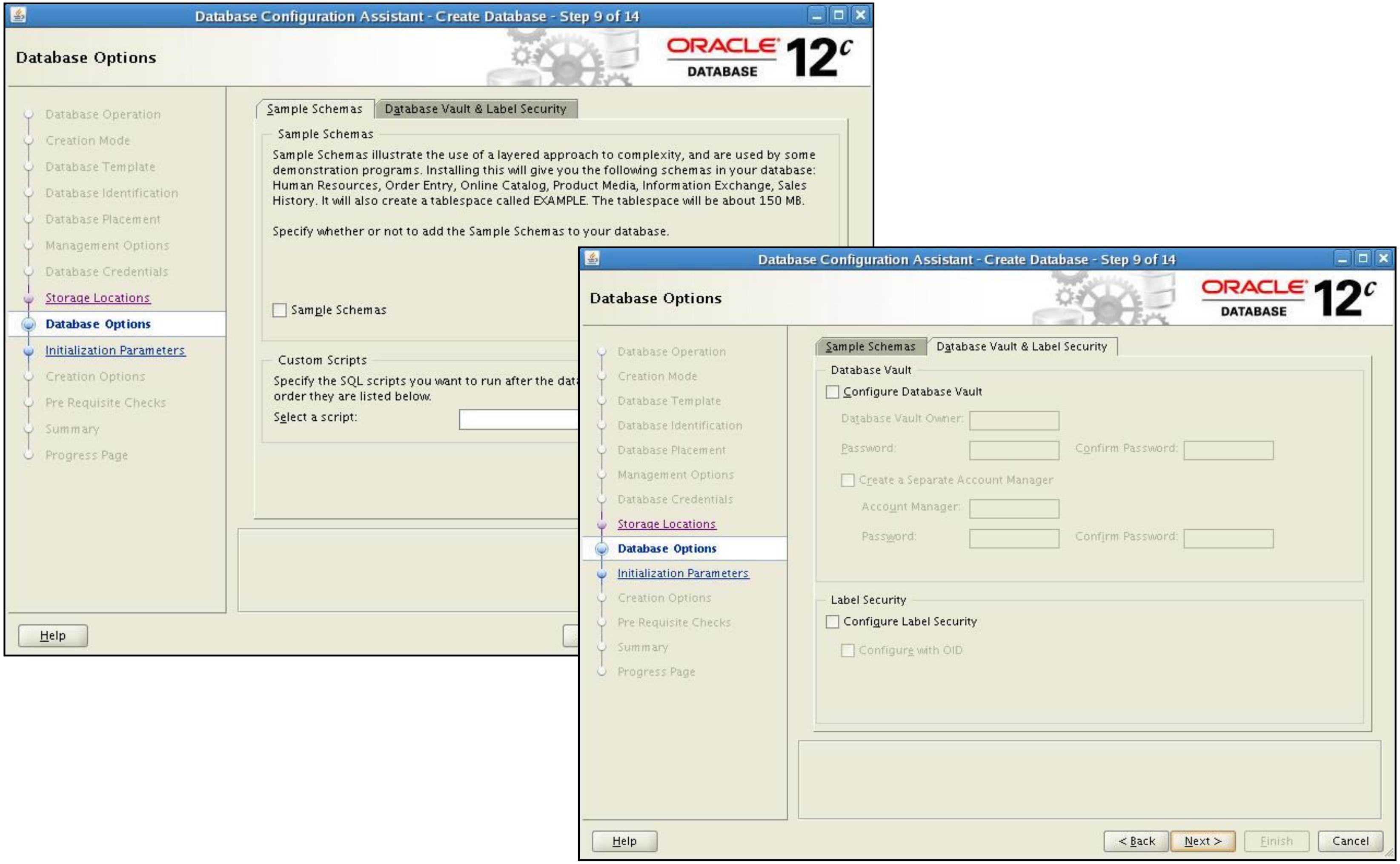
Passwords for Database Schema Owners



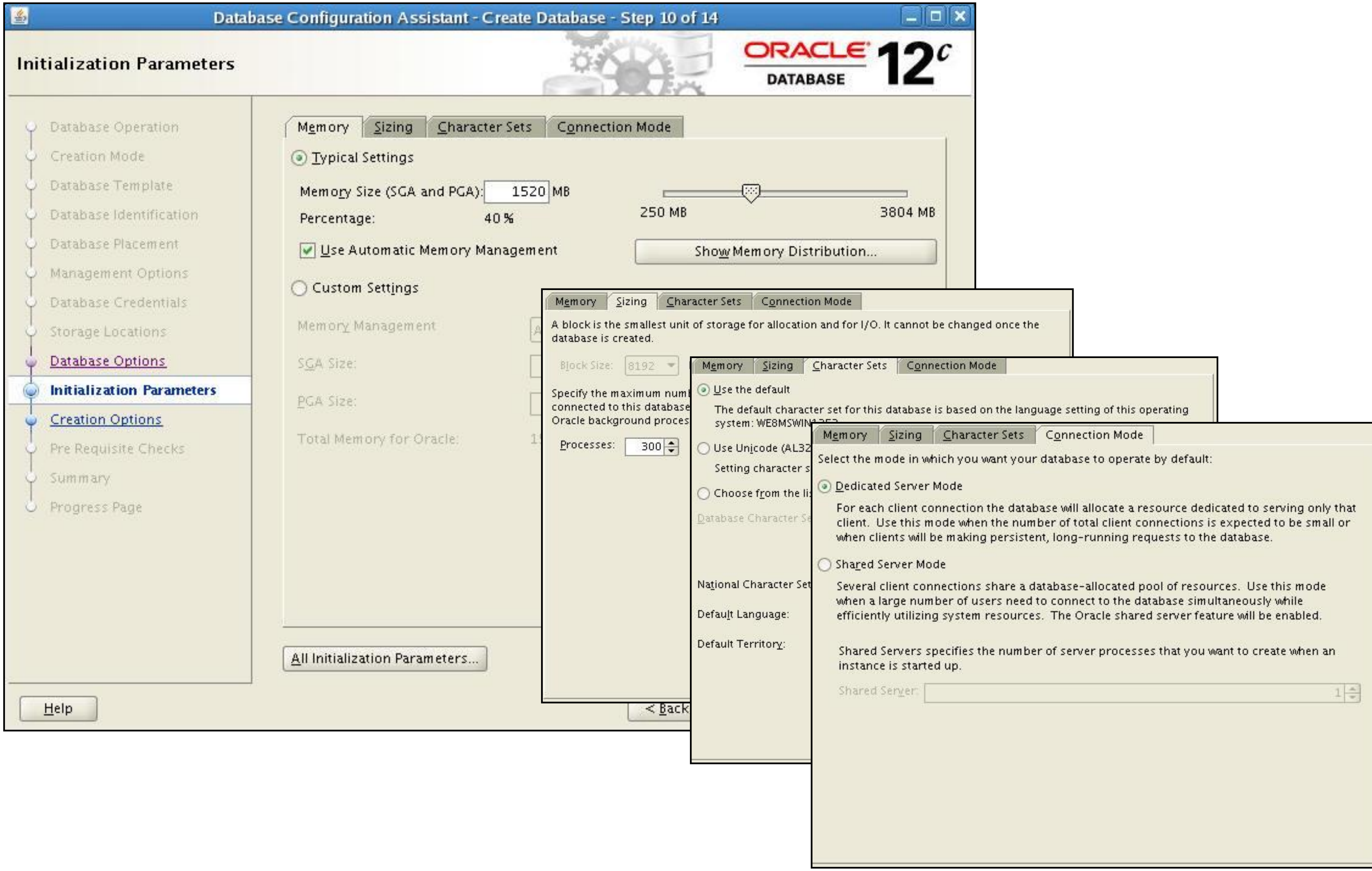
Storage Locations



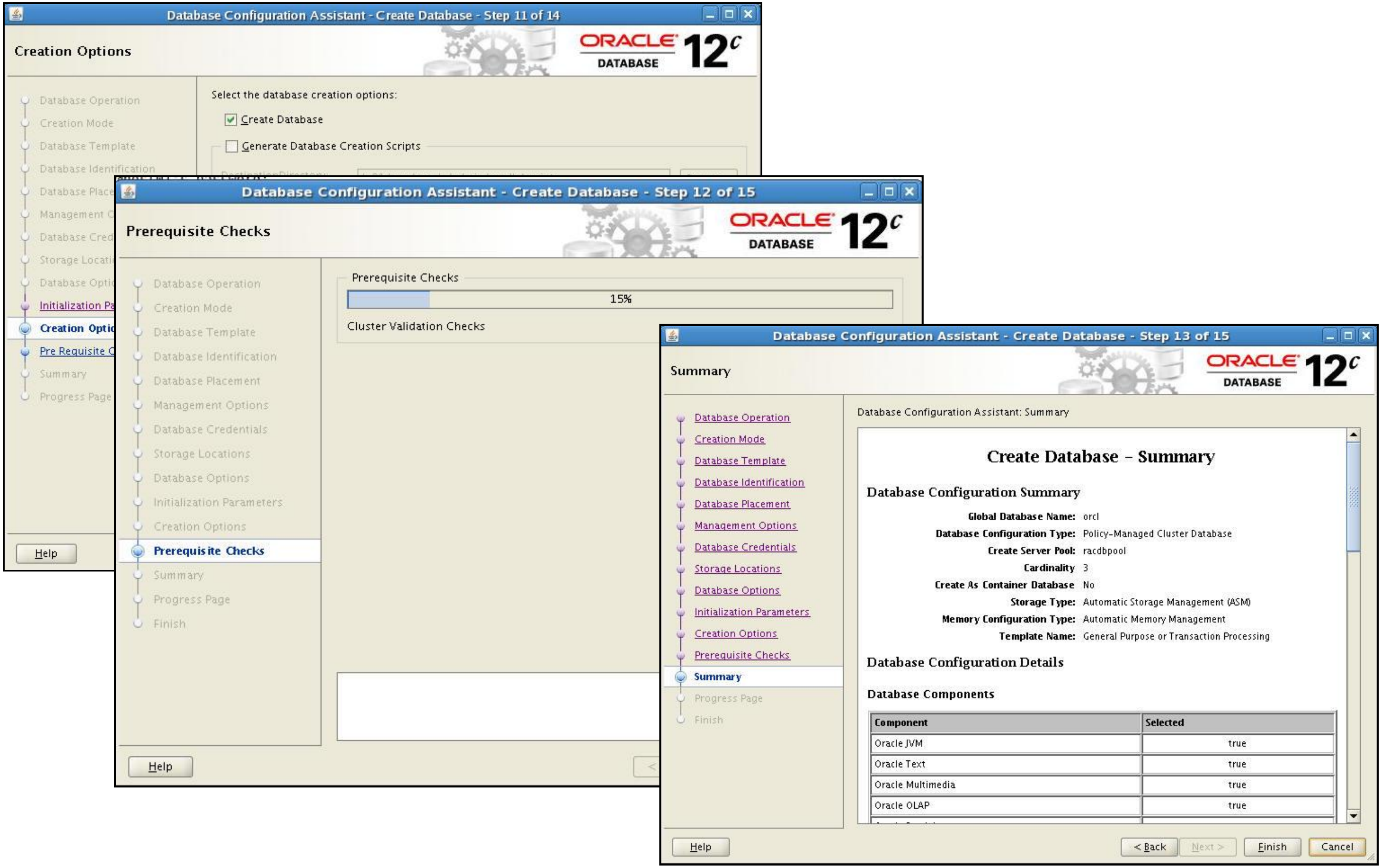
Database Content



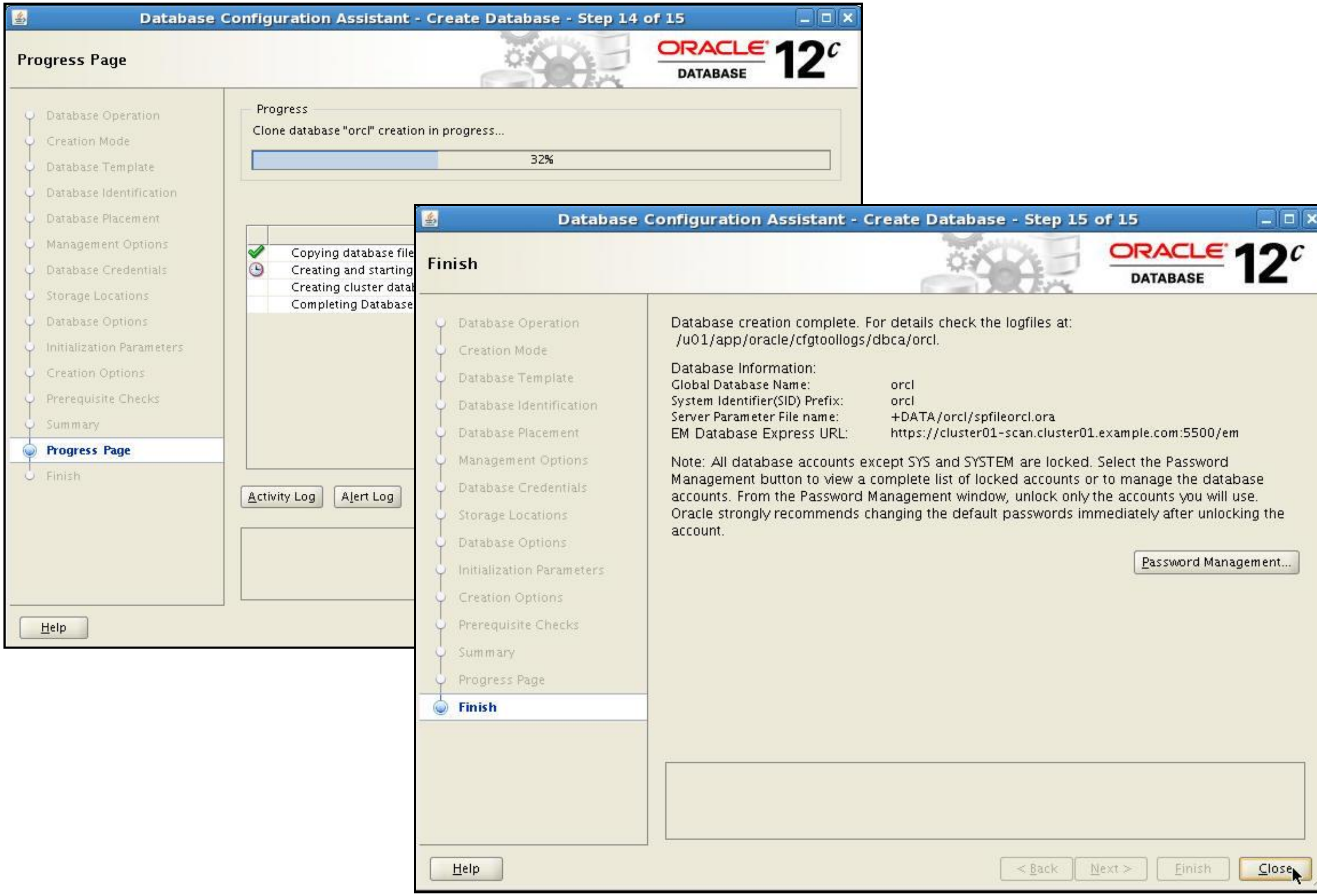
Initialization Parameters



Create the Database



Monitoring Progress



Postinstallation Tasks

- Download and install the required patch updates.
- Verify the cluster database configuration.

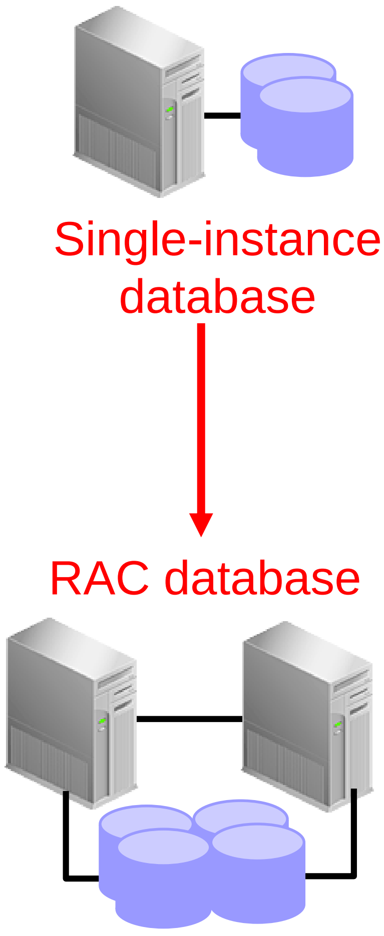
```
$ srvctl config database -db orcl
Database unique name: orcl
Database name: orcl
Oracle home: /u01/app/oracle/product/12.1.0/dbhome_1
Oracle user: oracle
Spfile: +DATA/ORCL/PARAMETERFILE/spfile.289.863634609
Password file: +DATA/ORCL/PASSWORD/pwdorcl.276.863633507
Domain: cluster01.example.com
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: orcldb
Disk Groups: DATA
Mount point paths:
Services:
Type: RAC
Start concurrency:
Stop concurrency:
OSDBA group: dba
OSOPER group: oper
Database instances:
Configured nodes:
Database is policy managed
```

Background Processes Specific to Oracle RAC

- `ACMS`: Atomic Control File to Memory Service
- `GTX[0-j]`: Global Transaction Process
- `LMON`: Global Enqueue Service Monitor
- `LMD`: Global Enqueue Service Daemon
- `LMS`: Global Cache Service Process
- `LCK0`: Instance Enqueue Process
- `LMHB`: Global Cache/Enqueue Service Heartbeat Monitor
- `PING`: Interconnect Latency Measurement Process
- `RCBG`: Result Cache Background Process
- `RMSn`: Oracle RAC Management Processes
- `RSMN`: Remote Slave Monitor

Single Instance–to-RAC Conversion

- Single-instance databases can be converted to RAC by using:
  - DBCA
  - Enterprise Manager
  - `RCONFIG` utility
- DBCA automates most of the conversion tasks.
- Before conversion, ensure that:
  - Your hardware and operating system are supported
  - Your cluster nodes have access to shared storage



Considerations for Converting Single-Instance Databases to Oracle RAC

- Backup procedures should be available before conversion takes place.
- Archiving in Oracle RAC environments requires a thread number in the archive file format.
- The archived logs from all instances of an Oracle RAC database are required for media recovery.
- By default, all database files are migrated to Oracle Managed Files (OMF).

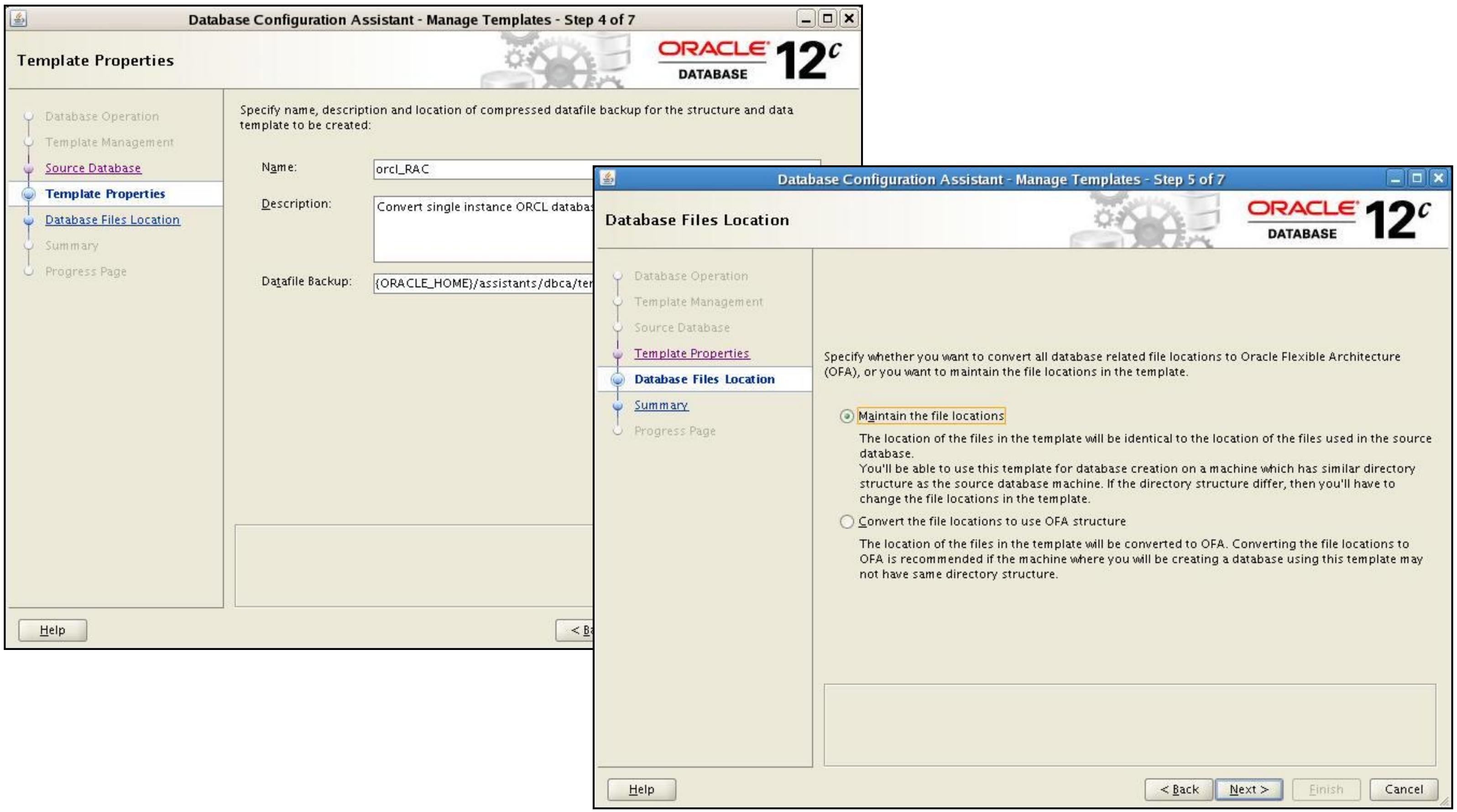
Single-Instance Conversion Using DBCA

Conversion steps for a single-instance database on *nonclustered* hardware:

1. Back up the original single-instance database.
2. Complete the Oracle Grid Infrastructure installation.
3. Validate the cluster.
4. Copy the preconfigured database image.
5. Install the Oracle Database 12*c* software with RAC.

Conversion Steps

1. Back up the original single-instance database.



Conversion Steps

2. Perform the preinstallation steps.
   – Tasks include kernel parameter configuration, hardware setup, network configuration, and shared storage setup.
3. Set up and validate the cluster.
   – Create a cluster with the required number of nodes according to your hardware vendor's documentation.
   – Validate cluster components before installation.
   – Install Oracle Clusterware.
   – Validate the completed cluster installation by using `cluvfy`.
4. Copy the preconfigured database image.
   – The database structure `*.dbc` file
   – The preconfigured database image `*.dfb` file
5. Install the Oracle Database 12*c* software with RAC.

Single-Instance Conversion Using `rconfig`

1. Locate the appropriate `.xml` file located in the `$ORACLE_HOME/assistants/rconfig/sampleXMLs` directory.
2. Modify the `ConvertToRAC_AdminManaged.xml` or `ConvertToRAC_PolicyManaged.xml` file as required for your system.
3. Save the file under a different name.

```
$ cd $ORACLE_HOME/assistants/rconfig/sampleXMLs
$ vi ConvertToRAC_PolicyManaged.xml
... Saved as my_rac_conversion.xml
$ rconfig my_rac_conversion.xml
```

Quiz

The RAC database software installation is initiated by executing `runInstaller` from the root directory of the Oracle Database 12*c* CD-ROM or from the software staging location.

a. True

b. False

Quiz

A single-instance database can be converted to a RAC database by using (choose the correct options):

a. `rconfig`
b. `netca`
c. `dbca`

Summary

In this lesson, you should have learned how to:

- Install the Oracle database software
- Create a cluster database
- Perform post-database-creation tasks
- Convert a single-instance Oracle database to RAC

Practice 3: Overview

This practice covers the following topics:

- Installing the Oracle database software
- Creating a RAC database

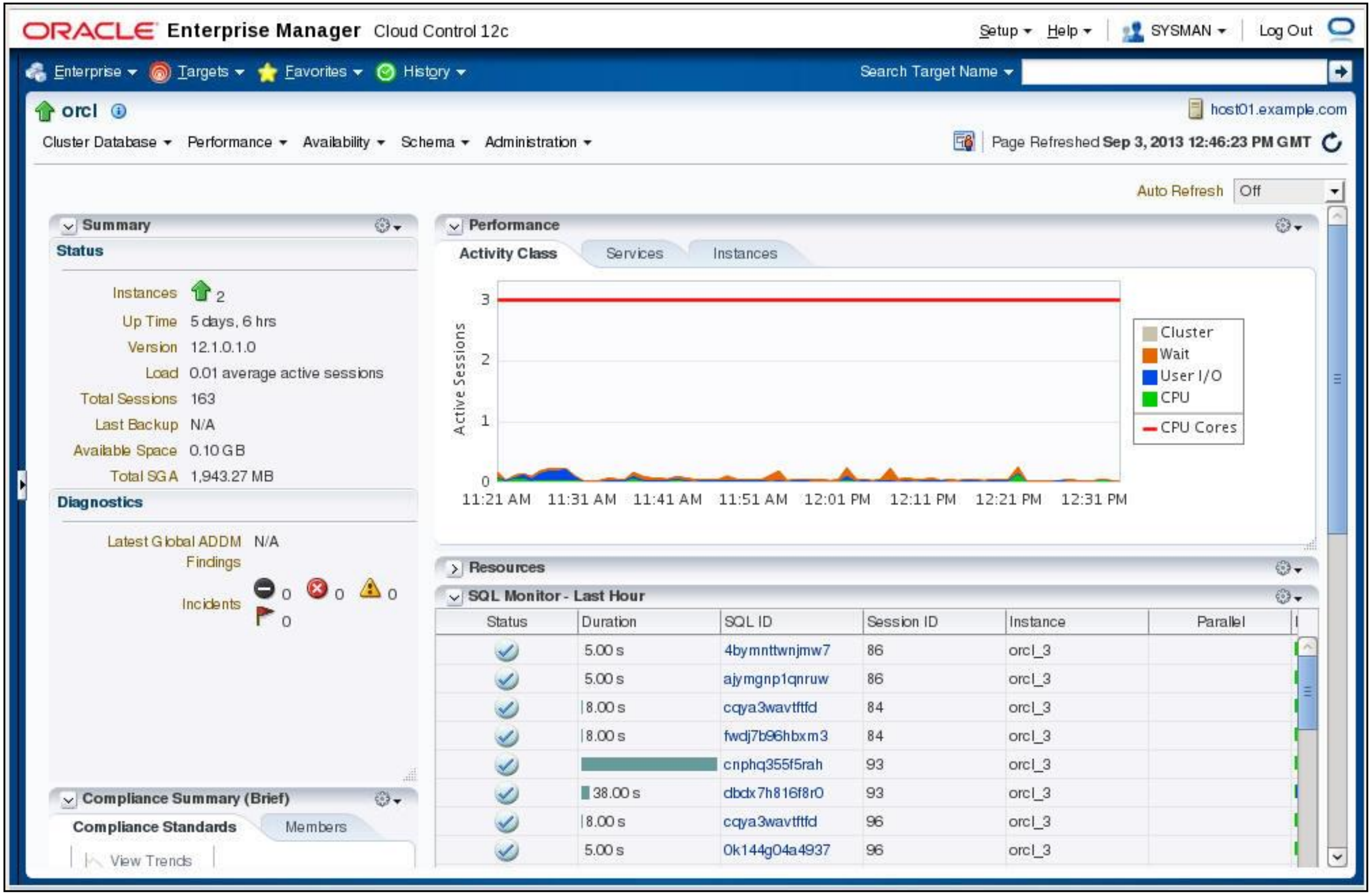
Oracle RAC Administration

Objectives

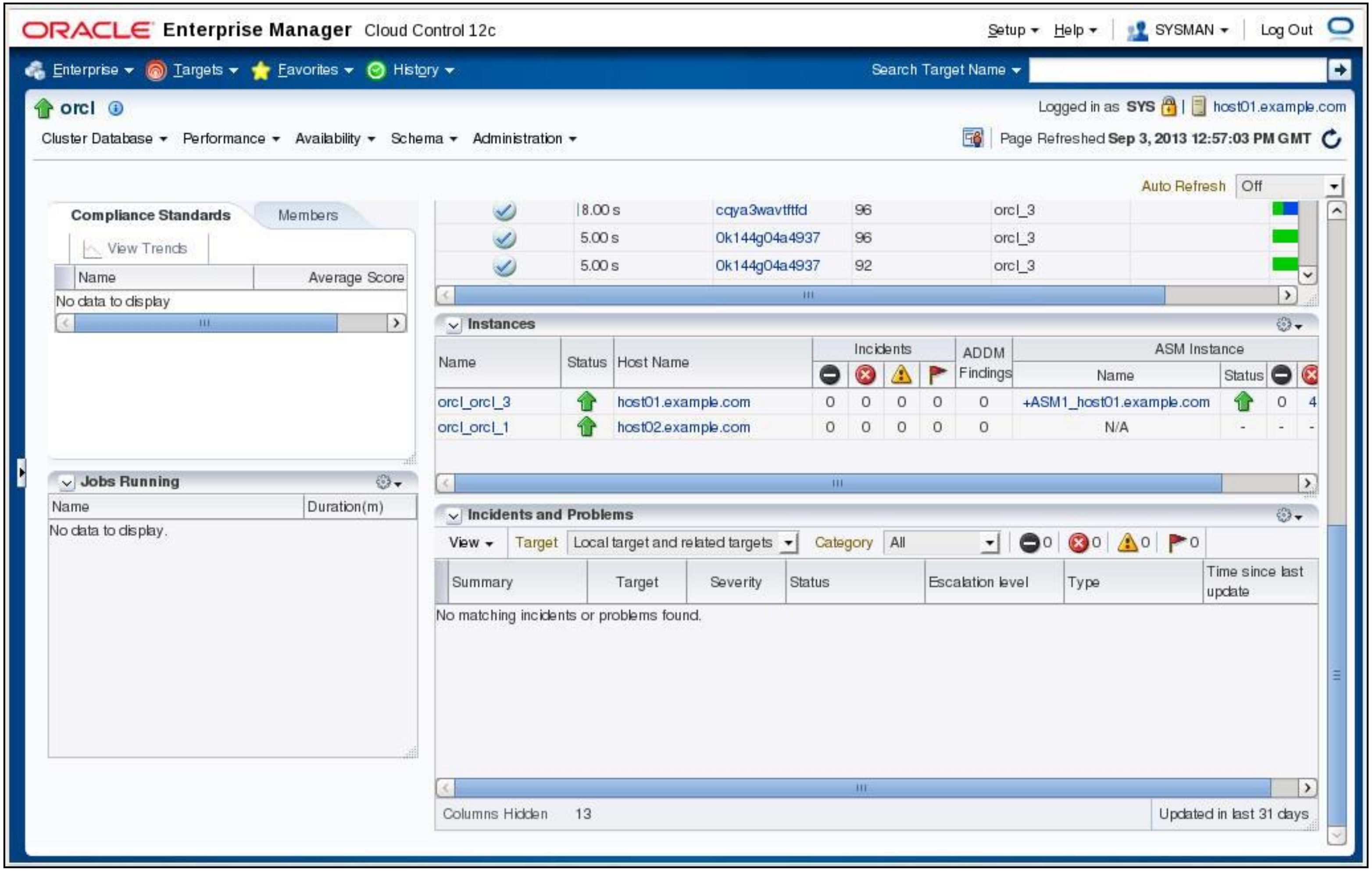
After completing this lesson, you should be able to:

- Use Enterprise Manager Cluster Database pages
- Define redo log files in a RAC environment
- Define undo tablespaces in a RAC environment
- Start and stop RAC databases and instances
- Modify initialization parameters in a RAC environment

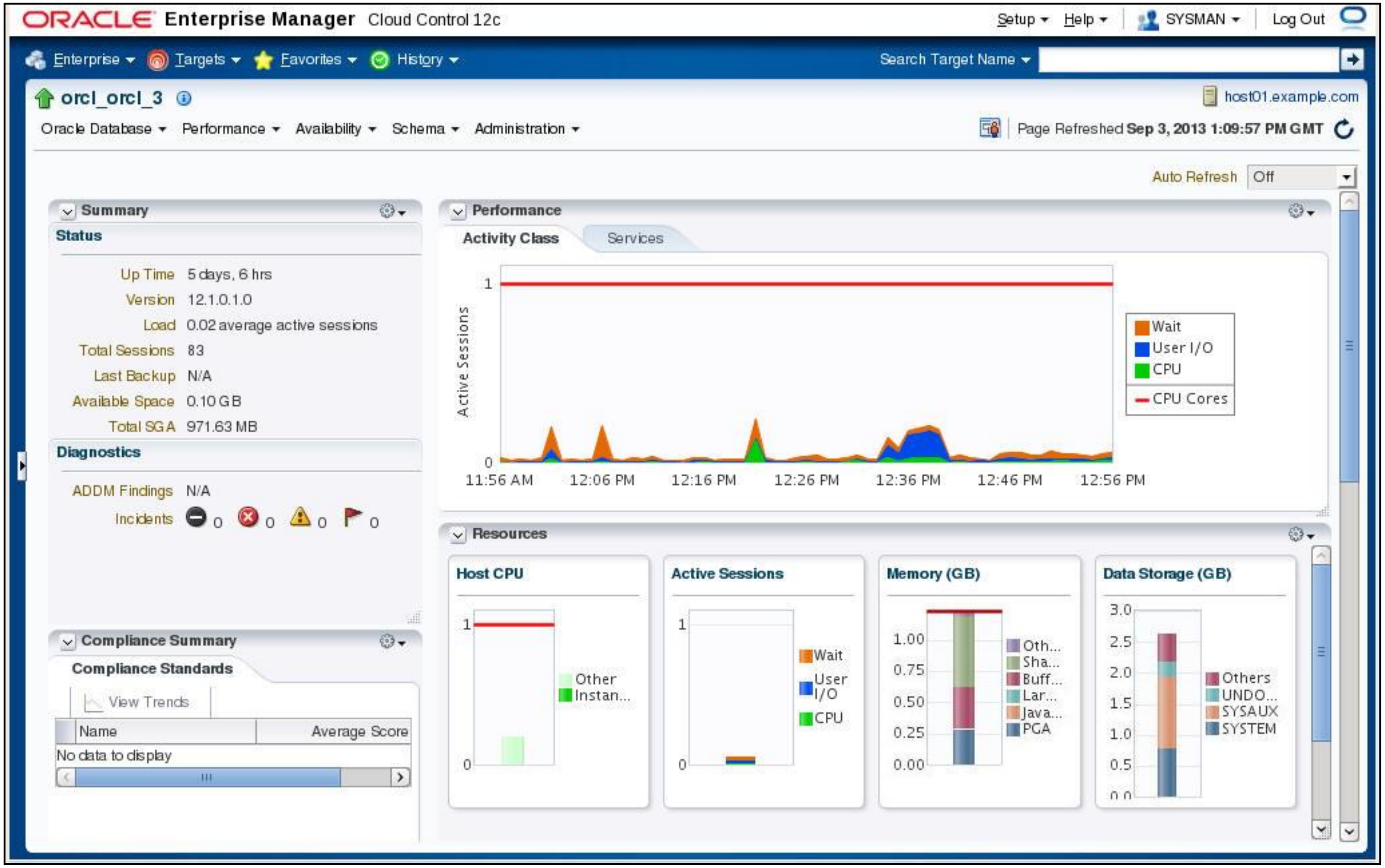
Enterprise Manager Cloud Control Cluster Database Home Page



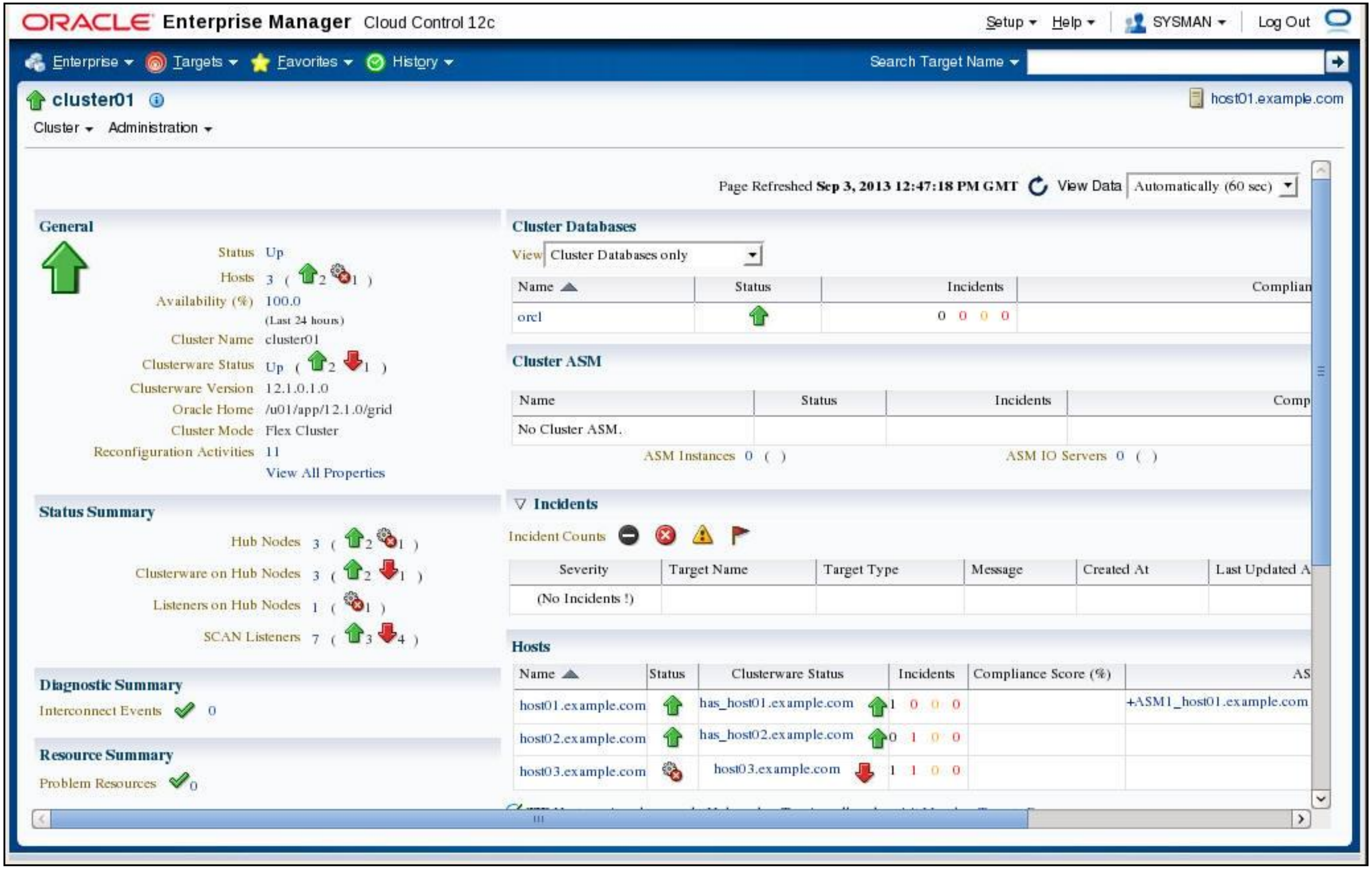
Cluster Database Home Page



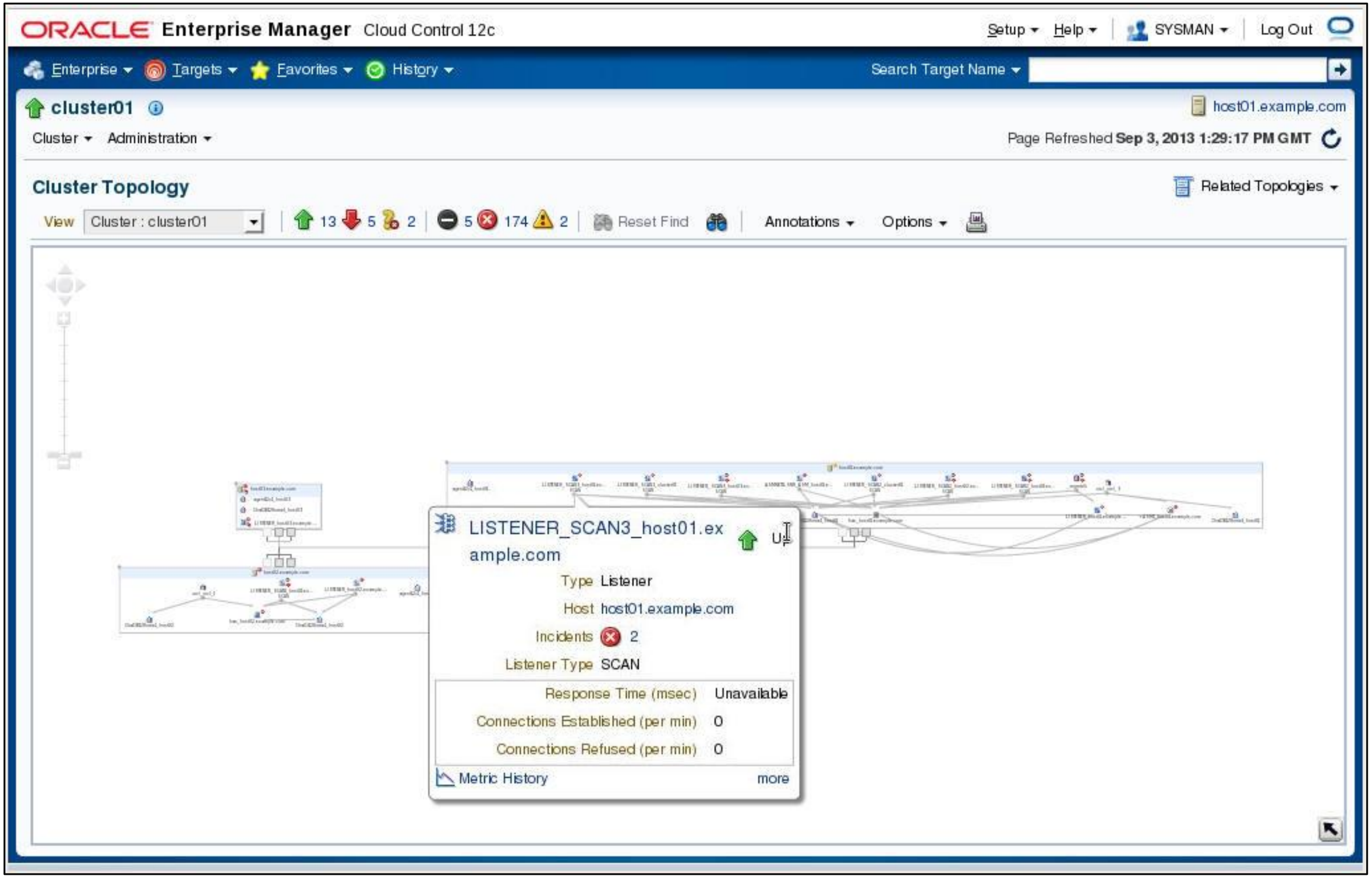
Cluster Database Instance Home Page



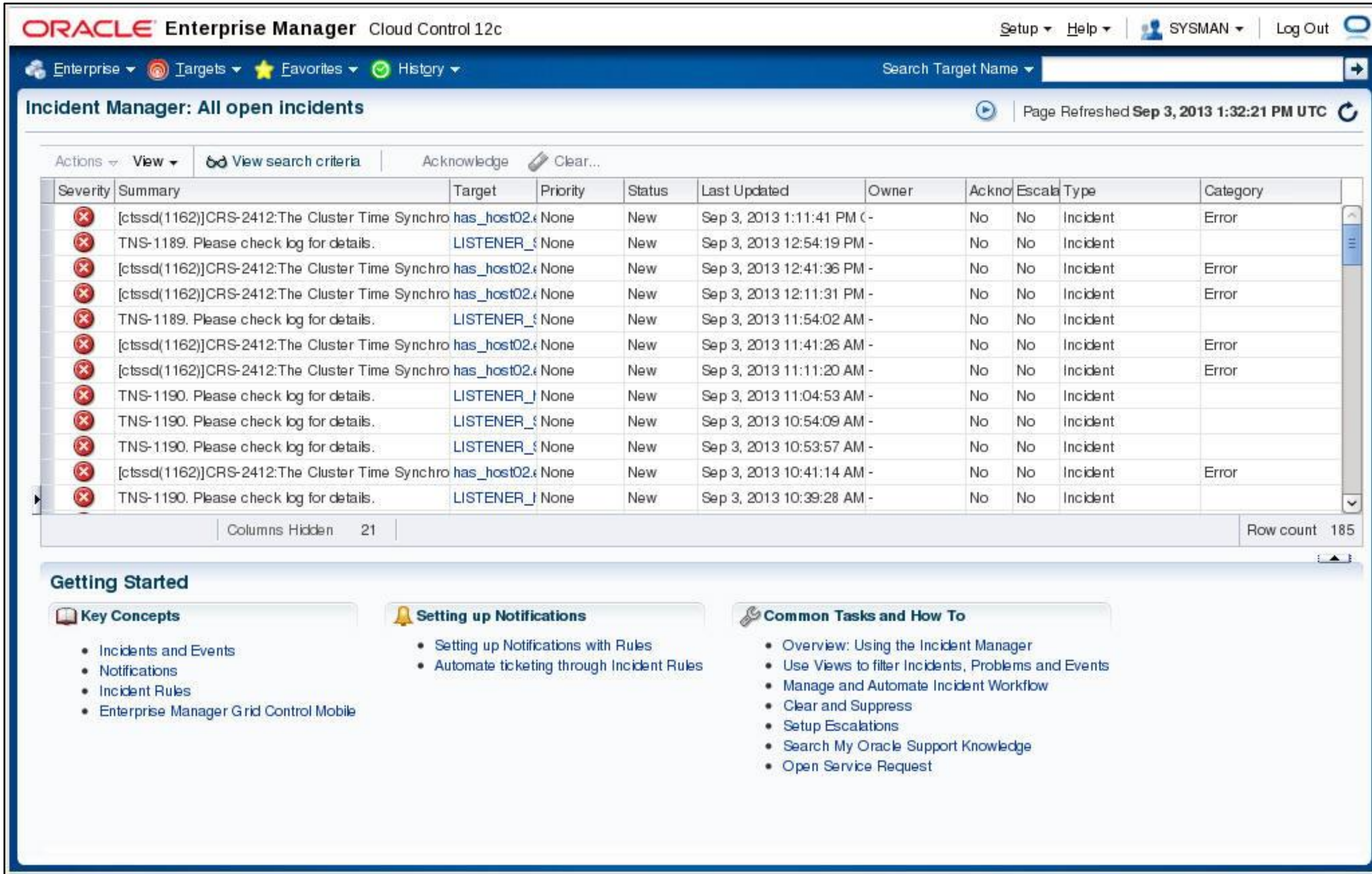
Cluster Home Page



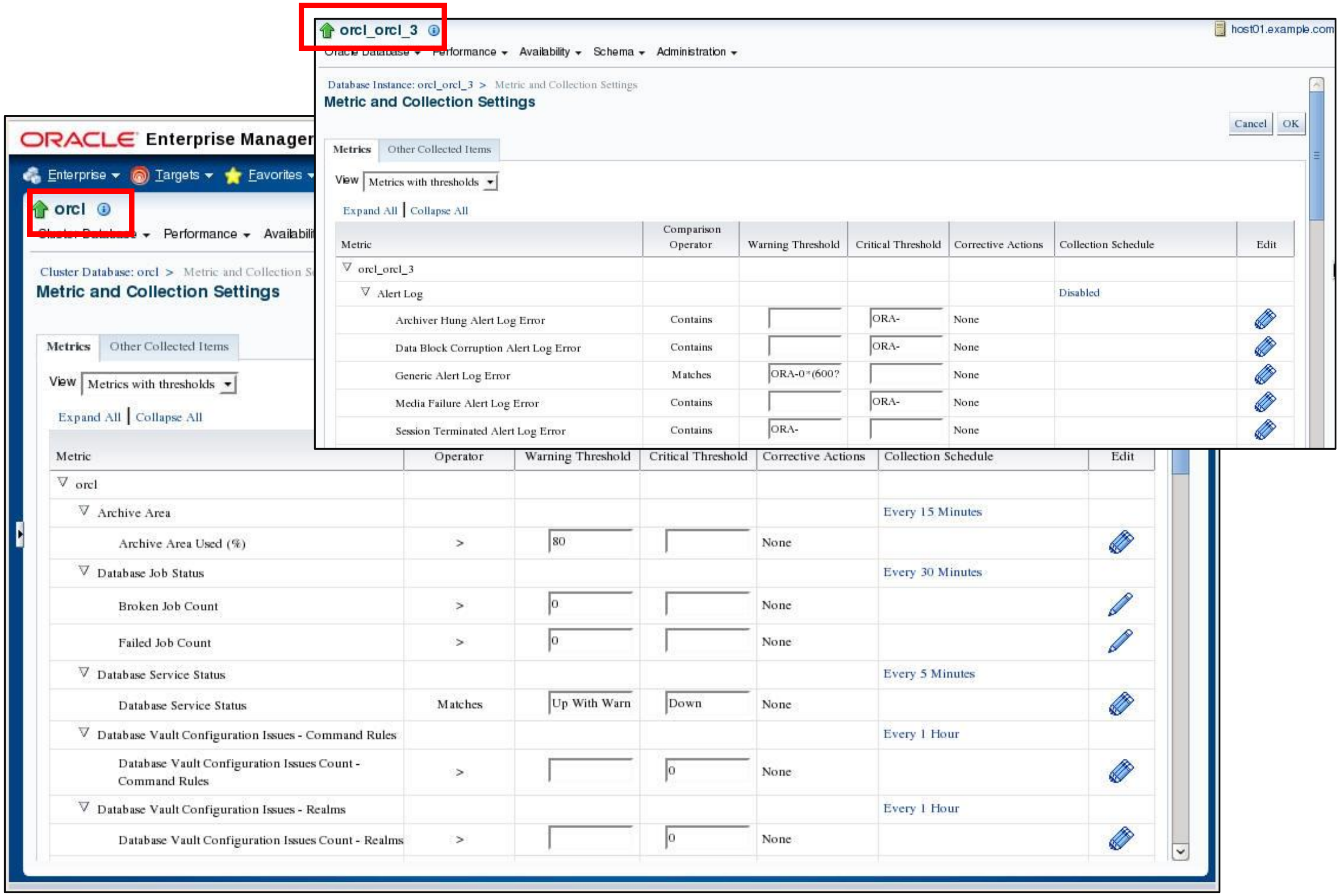
Topology Viewer



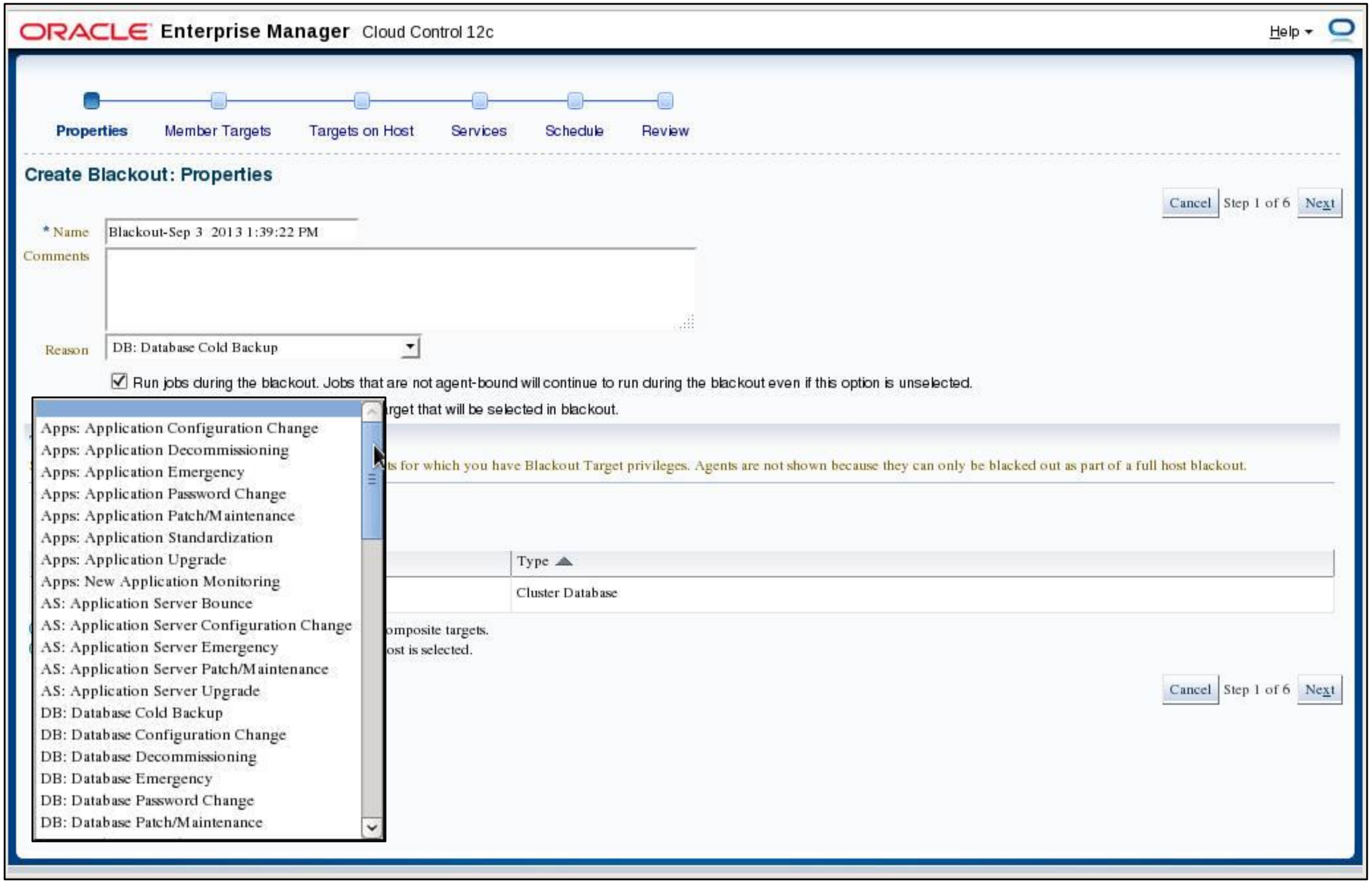
Enterprise Manager Alerts and RAC



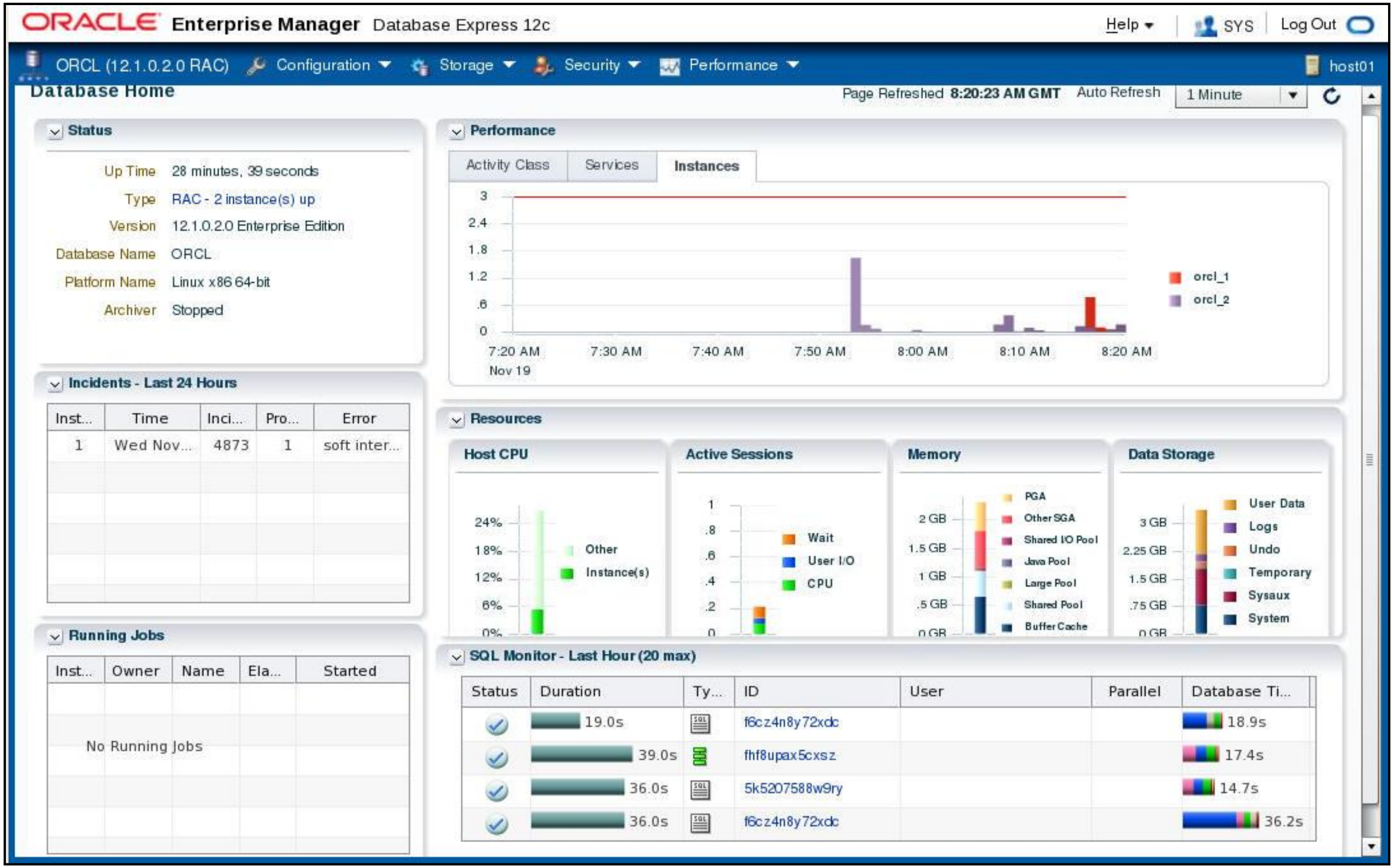
Enterprise Manager Metrics and RAC



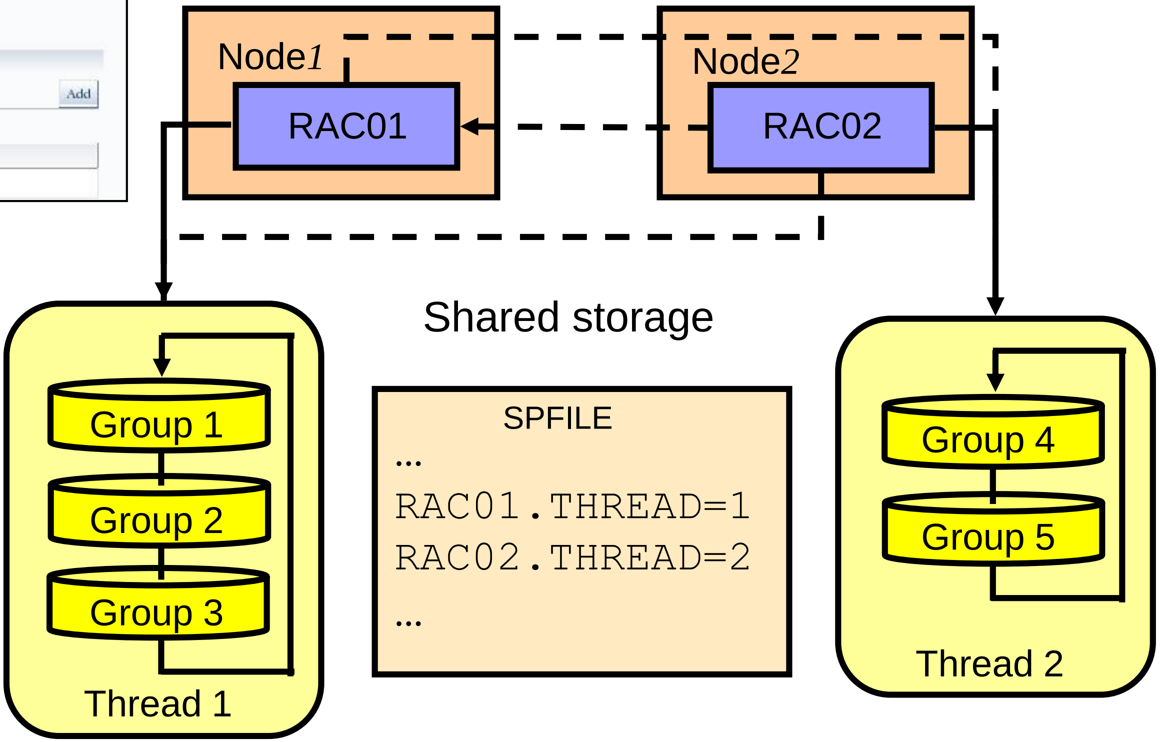
Enterprise Manager Blackouts and RAC



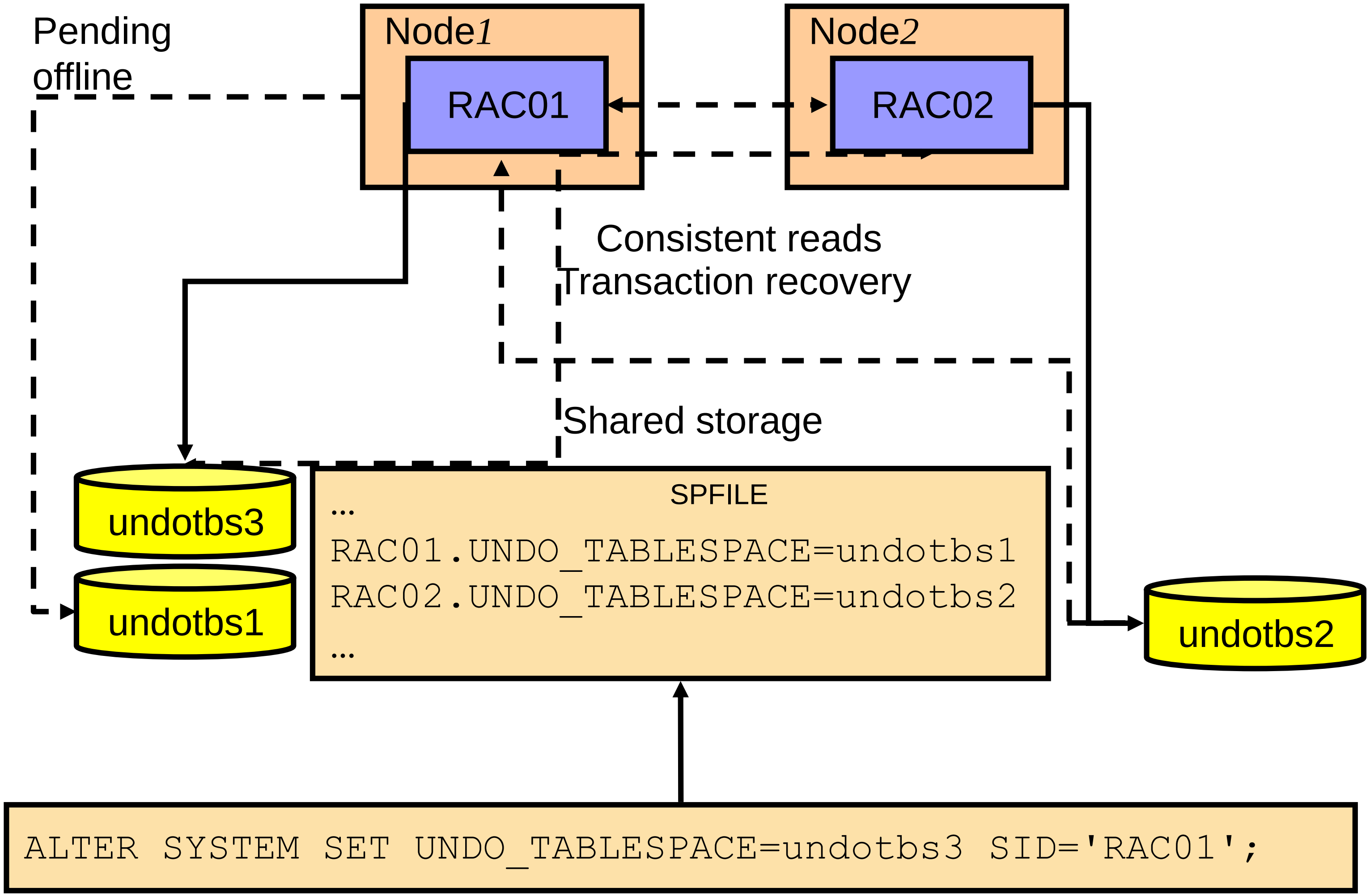
Enterprise Manager Database Express



Redo Log Files and RAC



Automatic Undo Management and RAC



Starting and Stopping RAC Instances

- Multiple instances can open the same database simultaneously.
- `SHUTDOWN TRANSACTIONAL LOCAL` does not wait for other instances' transactions to finish.
- RAC instances can be started and stopped by using:
  - Enterprise Manager
  - The Server Control (`srvctl`) utility
  - SQL*Plus
- Shutting down a RAC database means shutting down all instances accessing the database.

Starting and Stopping RAC Instances with `srvctl`

- `start/stop` syntax:

```
$ srvctl start instance -db db_unique_name –node node_name
-instance instance_name_list[-startoption
open|mount|nomount|normal|transactional|immediate|abort]
```

```
srvctl start|stop database -db <db_name> -eval
[-startoption
open|mount|nomount|normal|transactional|immediate|abort>]
```

- Examples:

```
$ srvctl start instance -db orcl -instance orcl1,orcl2
```

```
$ srvctl stop instance -db orcl -instance orcl1,orcl2
```

```
$ srvctl start database -db orcl -startoption open
```

```
$ srvctl start instance -db orcl -node host01

*** This command will start a Policy-Managed database****
```

Starting and Stopping RAC Instances with SQL*Plus

```
[host01] $ echo $ORACLE_SID
orcl1
sqlplus / as sysdba
SQL> startup
SQL> shutdown

[host02] $ echo $ORACLE_SID
orcl2
sqlplus / as sysdba
SQL> startup
SQL> shutdown
```

Starting and Stopping Pluggable Databases in Oracle RAC

- Manage RAC PDBs by managing services, regardless whether they are policy or administrator managed.
- Assign one database service to each PDB to coordinate start, stop, and placement of PDBs across instances.
  - Assume a PDB called `raccont` with a policy-managed PDB called `spark` in a server pool called `prod`:

```
$ srvctl add service -db raccont -pdb spark -service
plug -serverpool prod
```

- To start and stop the PDB:

```
$ srvctl start service -db raccont -service plug
```

```
$ srvctl stop service -db raccont -service plug
```

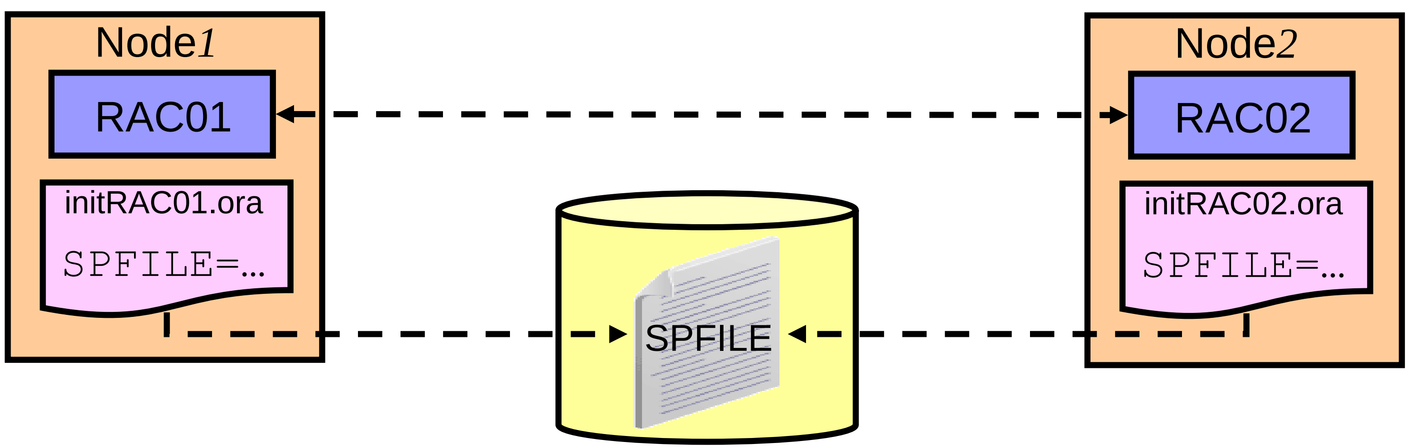
Switch Between Automatic and Manual Policies

```
$ srvctl config database -db orcl -a
Database unique name: orcl
Database name: orcl
Oracle home: /u01/app/oracle/product/12.1.0/dbhome_1
Oracle user: oracle
Spfile: +DATA/orcl/spfileorcl.ora
Password file: +DATA/orcl/orapworcl
Domain: cluster01.example.com
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: orcldb
Database instances:
Disk Groups:
Mount point paths:
Services:
Type: RAC
...
```

```
srvctl modify database -db orcl -policy MANUAL;
```

RAC Initialization Parameter Files

- An `SPFILE` is created if you use the DBCA.
- The `SPFILE` must be created in an ASM disk group or a cluster file system file.
- All instances use the same `SPFILE`.
- If the database is created manually, create an `SPFILE` from a `PFILE`.



SPFILE Parameter Values and RAC

- You can change parameter settings using the ALTER SYSTEM SET command from any instance:

```
ALTER SYSTEM SET <dpname> SCOPE=MEMORY sid='<sid|*>';
```

- SPFILE entries such as:
  - *.<pname> apply to all instances
  - <sid>.<pname> apply only to <sid>
  - <sid>.<pname> takes precedence over *.<pname>
- Use current or future *.<dpname> settings for <sid>:

```
ALTER SYSTEM RESET <dpname> SCOPE=MEMORY sid='<sid>';
```

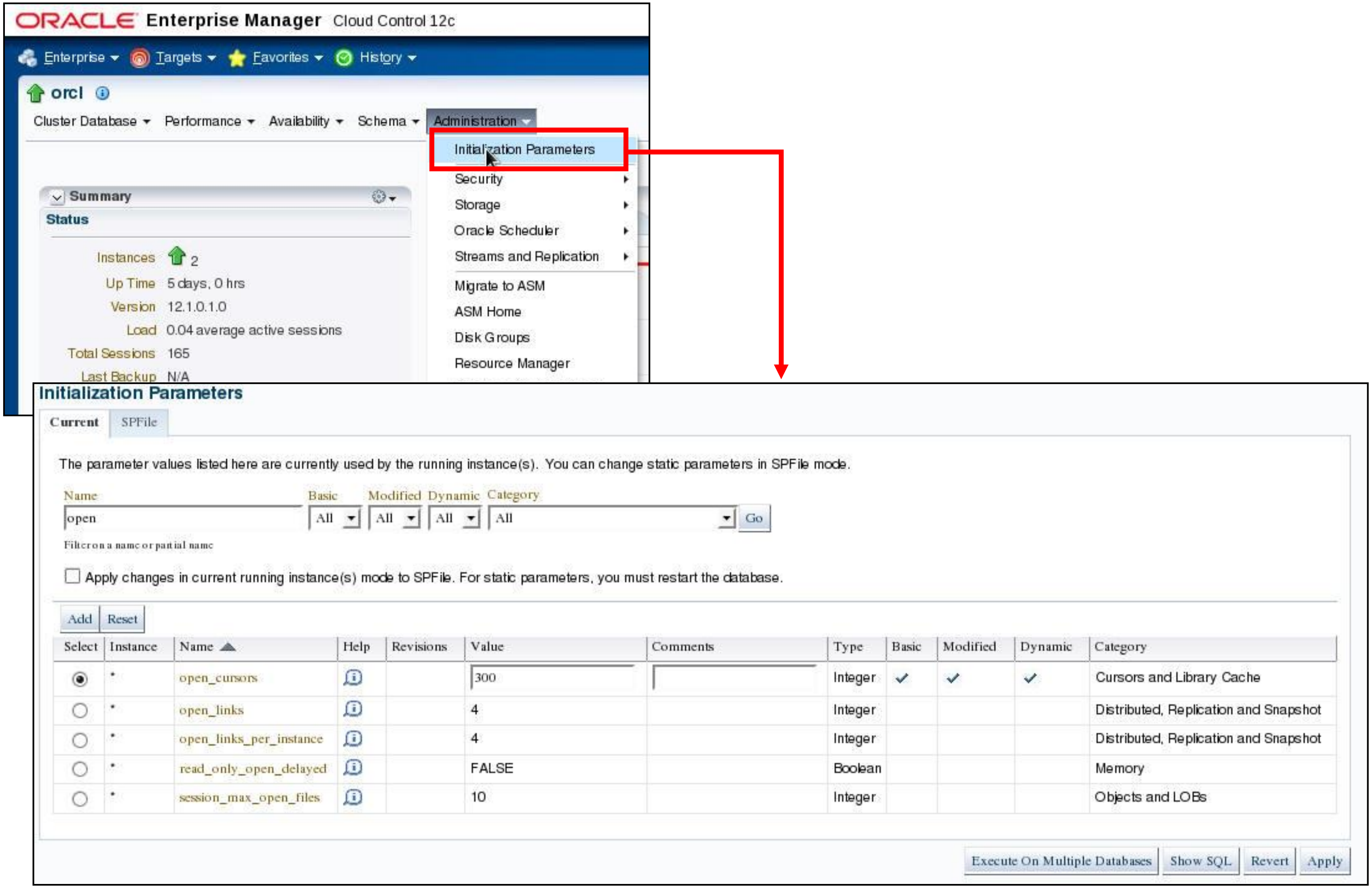
- Remove an entry from your SPFILE:

```
ALTER SYSTEM RESET <dpname> SCOPE=SPFILE sid='<sid|*>';
```

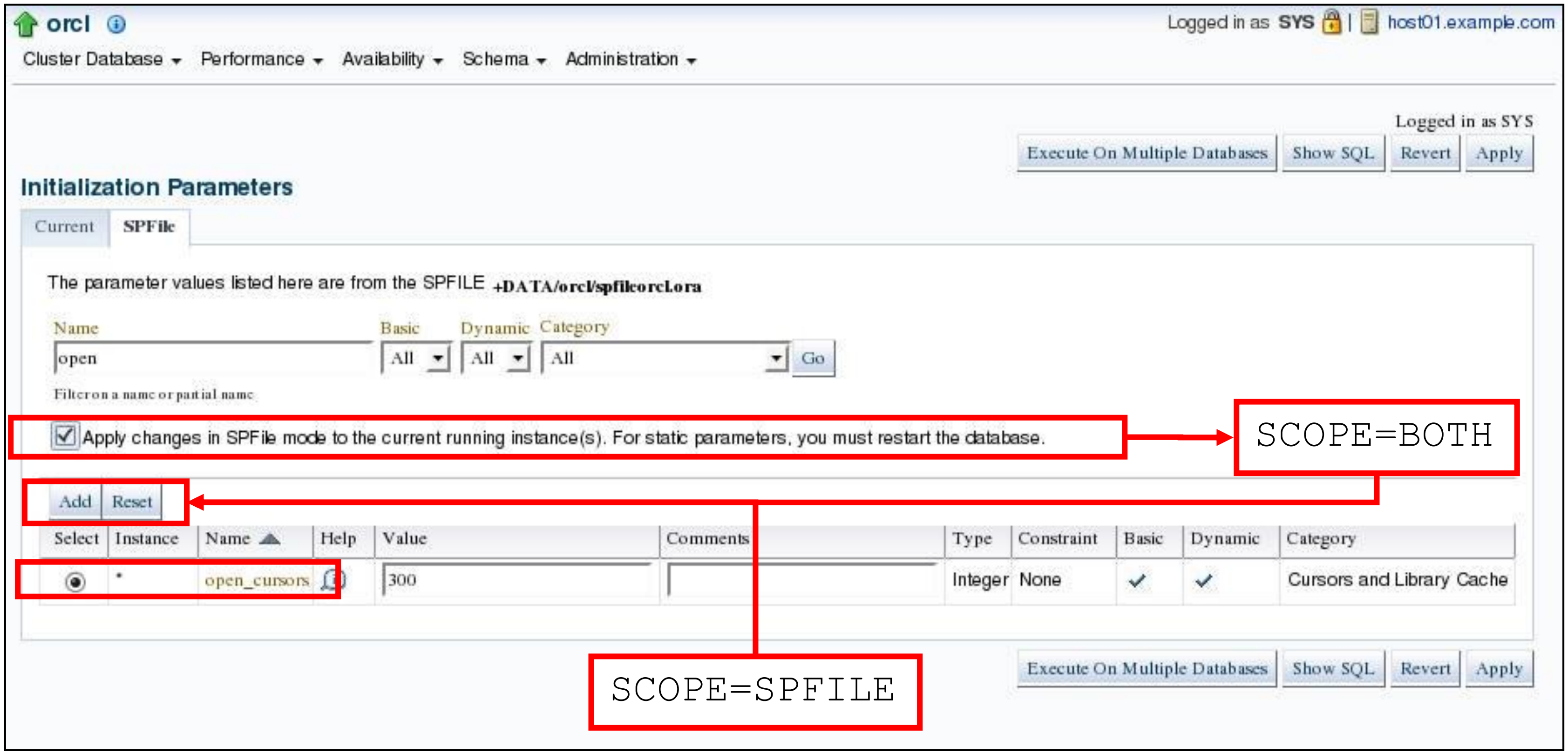
Parameter File Search Order in Oracle RAC

- On Linux and UNIX platforms, the search order is as follows:
  1. $ORACLE_HOME/dbs/spfilesid.ora
  2. $ORACLE_HOME/dbs/spfile.ora
  3. $ORACLE_HOME/dbs/initsid.ora
- On Windows platforms, the search order is as follows:
  1. %ORACLE_HOME%\database\spfilesid.ora
  2. %ORACLE_HOME%\database\spfile.ora
  3. %ORACLE_HOME%\database\initsid.ora

EM and `SPFILE` Parameter Values



EM and SPFILE Parameter Values



RAC Initialization Parameters

Initialization Parameters

Current | SPFile

The parameter values listed here are currently used by the running instance(s). You can change static parameters in SPFile mode.

Name: cluster | Basic: All | Modified: All | Dynamic: All | Category: All | Go

Filter on a name or partial name

☑ Apply changes in current running instance(s) mode to SPFile. For static parameters, you must restart the database.

Add | Reset

Select	Instance	Name	Help	Revisions	Value	Comments	Type	Basic	Modified	Dynamic	Category
◉	*	cluster_database			TRUE		Boolean	✓	✓		Cluster Database
○	*	cluster_database_instances			3		Integer				Cluster Database
○	*	cluster_interconnects					String				Cluster Database

Execute On Multiple Databases | Show SQL | Revert | Apply

◉	*	db_name			orcl		String	✓	✓		Database Identification

◉	*	dispatchers			(PROTOCOL=TCP) (SERV		String		✓	✓	Shared Server

◉	*	spfile			+DATA/orcl/spfileorcl.ora		String			✓	Miscellaneous

◉	orcl_1	instance_name			orcl_1		String				Instance Identification
○	orcl_3	instance_name			orcl_3		String				Instance Identification

Parameters That Require Identical Settings

- COMPATIBLE
- CLUSTER_DATABASE
- CONTROL_FILES
- DB_BLOCK_SIZE
- DB_DOMAIN
- DB_FILES
- DB_NAME
- DB_RECOVERY_FILE_DEST
- DB_RECOVERY_FILE_DEST_SIZE
- DB_UNIQUE_NAME
- INSTANCE_TYPE (RDBMS or ASM)
- PARALLEL_EXECUTION_MESSAGE_SIZE
- REMOTE_LOGIN_PASSWORDFILE
- UNDO_MANAGEMENT

Parameters That Require Unique Settings

Instance settings:

- `INSTANCE_NAME`
- `INSTANCE_NUMBER`
- `UNDO_TABLESPACE`
- `CLUSTER_INTERCONNECTS`
- `ROLLBACK_SEGMENTS`

Quiescing RAC Databases

- Use the `ALTER SYSTEM QUIESCE RESTRICTED` statement from a single instance:

```
SQL> ALTER SYSTEM QUIESCE RESTRICTED;
```

- You must have the Database Resource Manager feature activated to issue the preceding statement.
- The database cannot be opened by other instances after the `ALTER SYSTEM QUIESCE…` statement starts.
- The `ALTER SYSTEM QUIESCE RESTRICTED` and `ALTER SYSTEM UNQUIESCE` statements affect all instances in a RAC environment.
- Cold backups cannot be taken when the database is in a quiesced state.

Terminating Sessions on a Specific Instance

```
SQL> SELECT SID, SERIAL#, INST_ID
  2  FROM GV$SESSION WHERE USERNAME='JMW';

       SID    SERIAL#    INST_ID
---------- ---------- ----------
       140       3340          2

SQL> ALTER SYSTEM KILL SESSION '140,3340,@2';

System altered.

SQL>
```

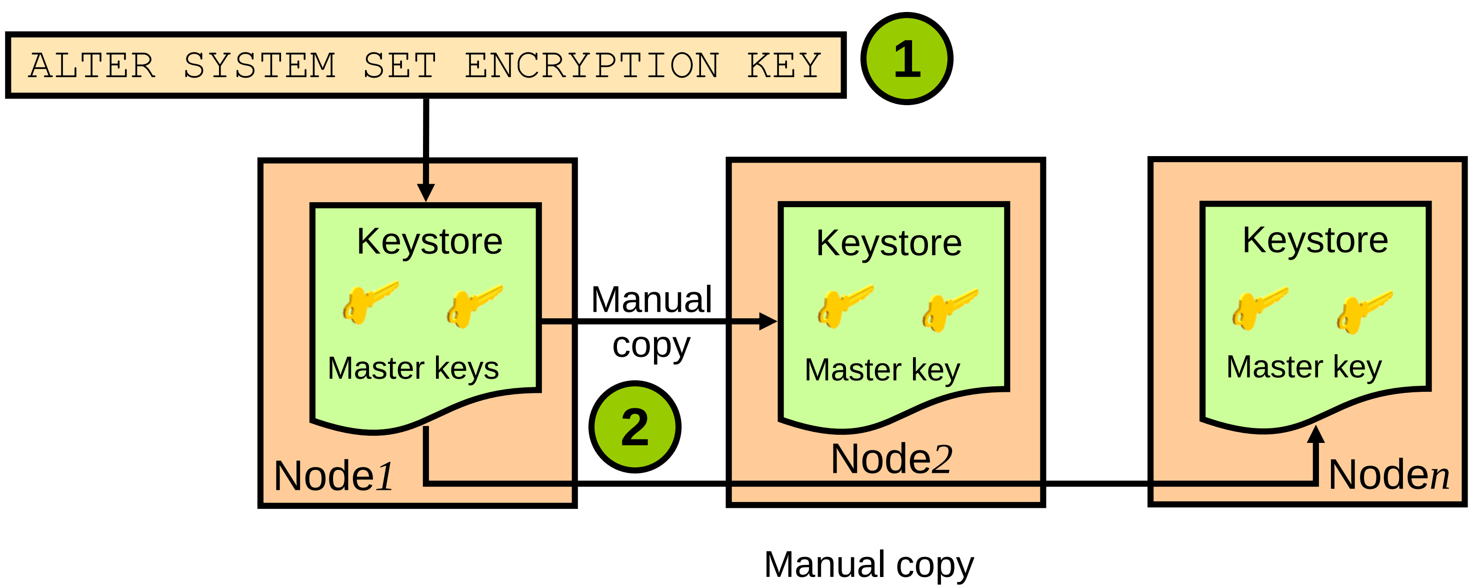
```
ALTER SYSTEM KILL SESSION '140,3340,@2'
*
ERROR at line 1:

ORA-00031: session marked for kill
```

How SQL*Plus Commands Affect Instances

SQL*Plus Command	Associated Instance
ARCHIVE LOG	Generally affects the current instance
CONNECT	Affects the default instance if no instance is specified in the CONNECT command
RECOVER	Does not affect any particular instance, but rather the database
SHOW PARAMETER and SHOW SGA	Show the current instance parameter and SGA information
STARTUP and SHUTDOWN	Affect the current instance
SHOW INSTANCE	Displays information about the current instance

Transparent Data Encryption and Keystores in RAC

- One wallet shared by all instances on shared storage:
  - No additional administration is required.
- One copy of the keystore on each local storage:
  - Local copies need to be synchronized each time master key is changed.



Quiz

If an instance starts in a policy-managed RAC environment and no thread or redo log file is available, then Oracle Clusterware automatically enables a thread of redo and allocates the redo log files and undo if the database uses Oracle ASM or any cluster file system and OMF is enabled.

a. True
b. False

Quiz

Which of the following statements is *not* true?

a. Multiple instances can open the same database simultaneously.

b. Shutting down one instance does not interfere with other running instances.

c. `SHUTDOWN TRANSACTIONAL LOCAL` will wait for other instances' transactions to finish.

d. Shutting down a RAC database means shutting down all instances accessing the database.

Summary

In this lesson, you should have learned how to:

- Use Enterprise Manager Cluster Database pages
- Define redo log files in a RAC environment
- Define undo tablespaces in a RAC environment
- Start and stop RAC databases and instances
- Modify initialization parameters in a RAC environment

Practice 4: Overview

This practice covers the following topics:

- Using operating system– and password file–authenticated connections
- Using Oracle Database authenticated connections
- Stopping a complete ORACLE_HOME component stack

Upgrading and Patching Oracle RAC

Objectives

After completing this lesson, you should be able to:

- Describe the types of patches available
- Plan for rolling patches and rolling upgrades
- Install a patch set with the Oracle Universal Installer (OUI) utility
- Install a patch with the `opatch` utility

Patch and Patch Set: Overview

- Oracle issues product fixes for its software called patches.
- Patches are associated with particular releases and versions of Oracle products.
- The patching cycle involves downloading patches, applying patches, and verifying the applied patch.
- Patching involves migrating from one version of the software product to another, within a particular release.
- When a patch is applied to an Oracle software installation, it updates the executable files, libraries, and object files in the software home directory.
  - The patch application can also update configuration files and Oracle-supplied SQL schemas.

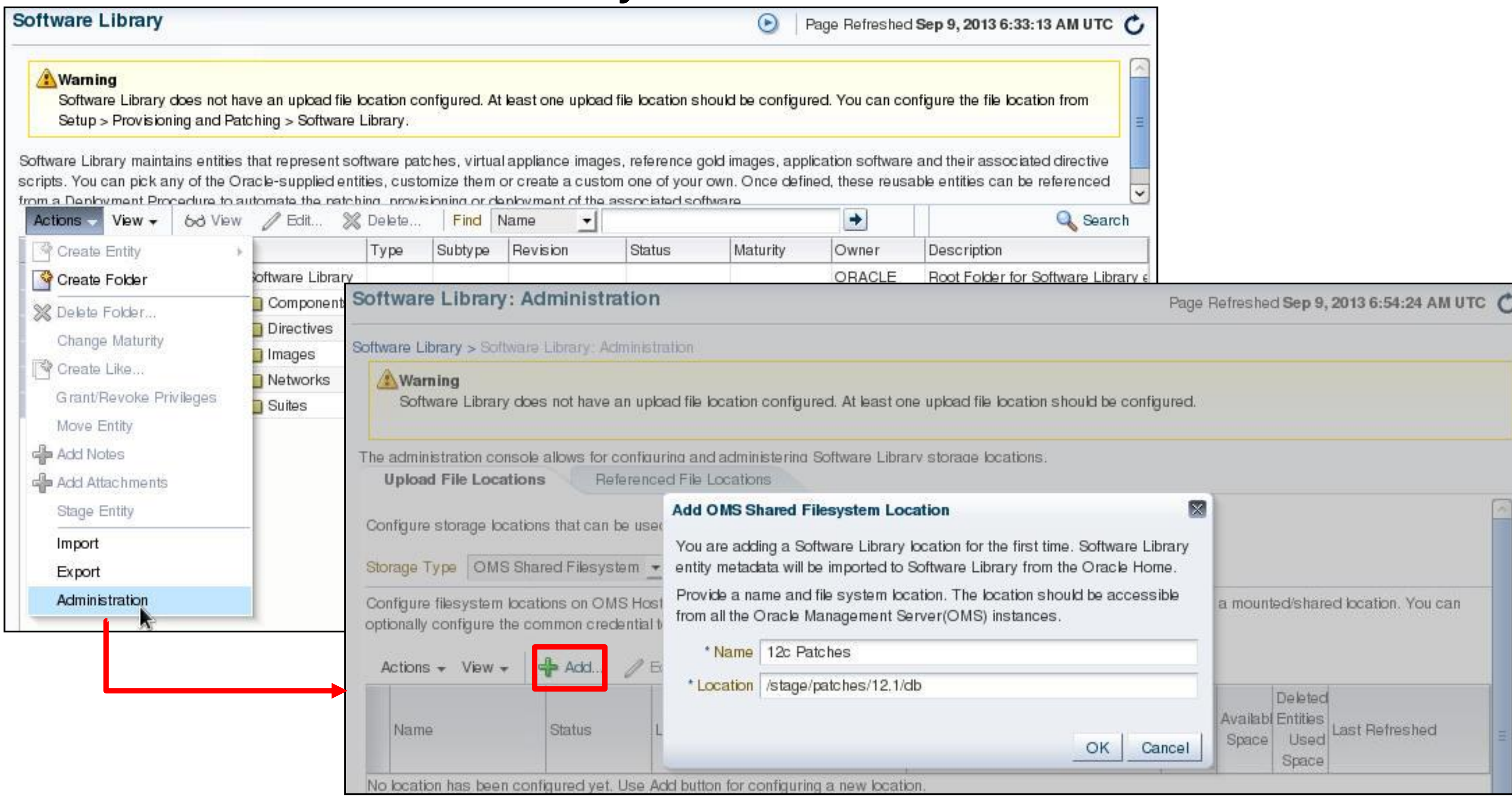
Types of Patches

Patch Type	Description
Interim Patches	Released to fix a bug, or a collection of bugs. Previously called patch set exceptions (PSE), one-off patches, or hot fixes.
Interim Patches (for Security bug fixes)	Released to provide customer-specific security fixes. Previously referred to as a test patch, fix verification binary, or e-fix.
Diagnostic Patches	Mainly help diagnose and verify a fix, or a collection of bugfixes
Bundle Patch Updates	Cumulative collection of fixes for a specific product or component. Previously referred to as a maintenance pack, service pack, cumulative patch, update release, or MLR.
Patch Set Updates (PSU)	Cumulative patch bundles that contain well-tested and proven bug fixes for critical issues. PSUs have limited new content, and do not include any changes that require re-certification.
Security Patch Updates	A cumulative collection of security bug fixes

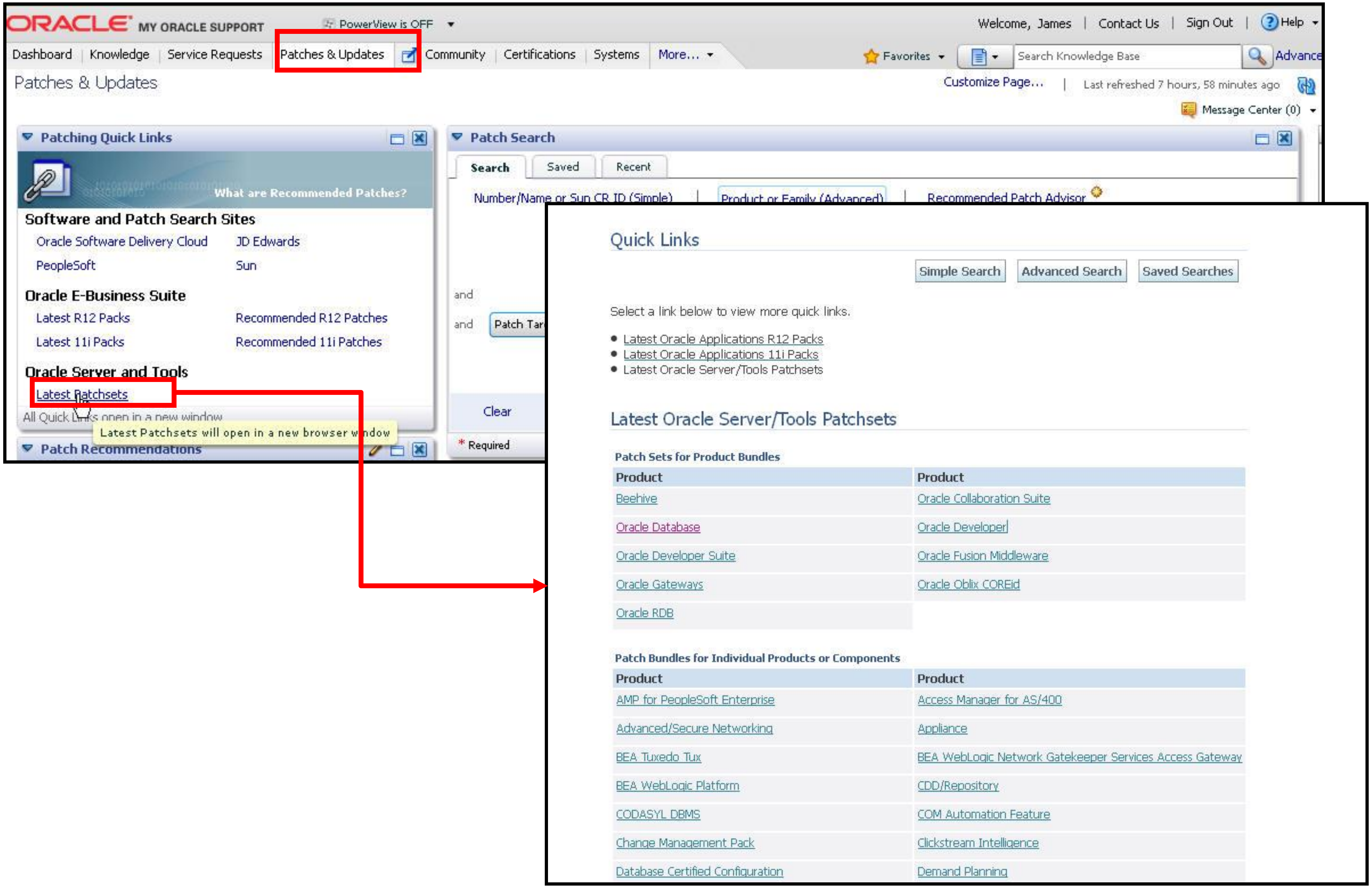
Configuring the Software Library

Use the Provisioning page of Enterprise Manager.

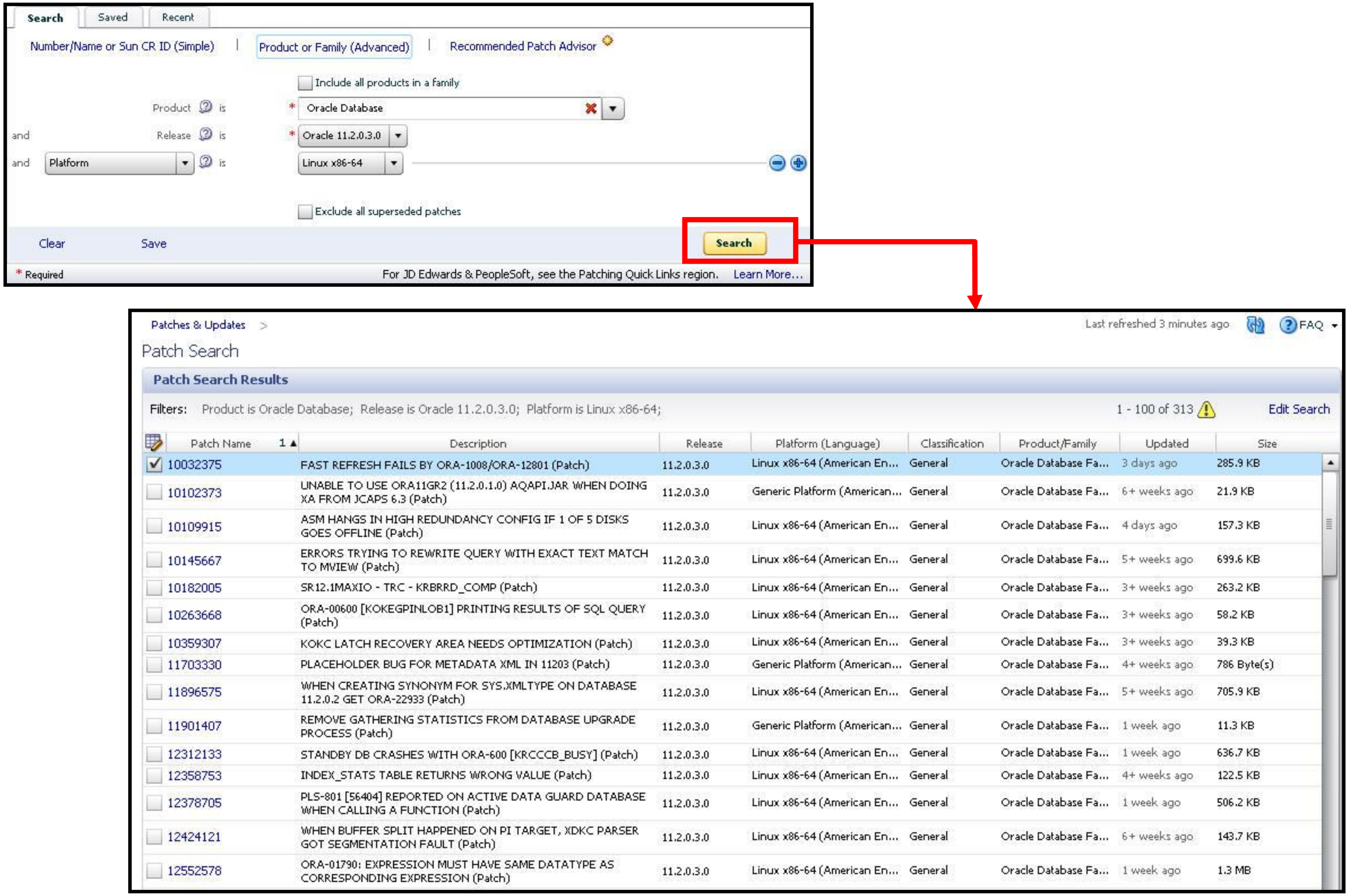
- Add a software library location:



Obtaining Oracle RAC Patches

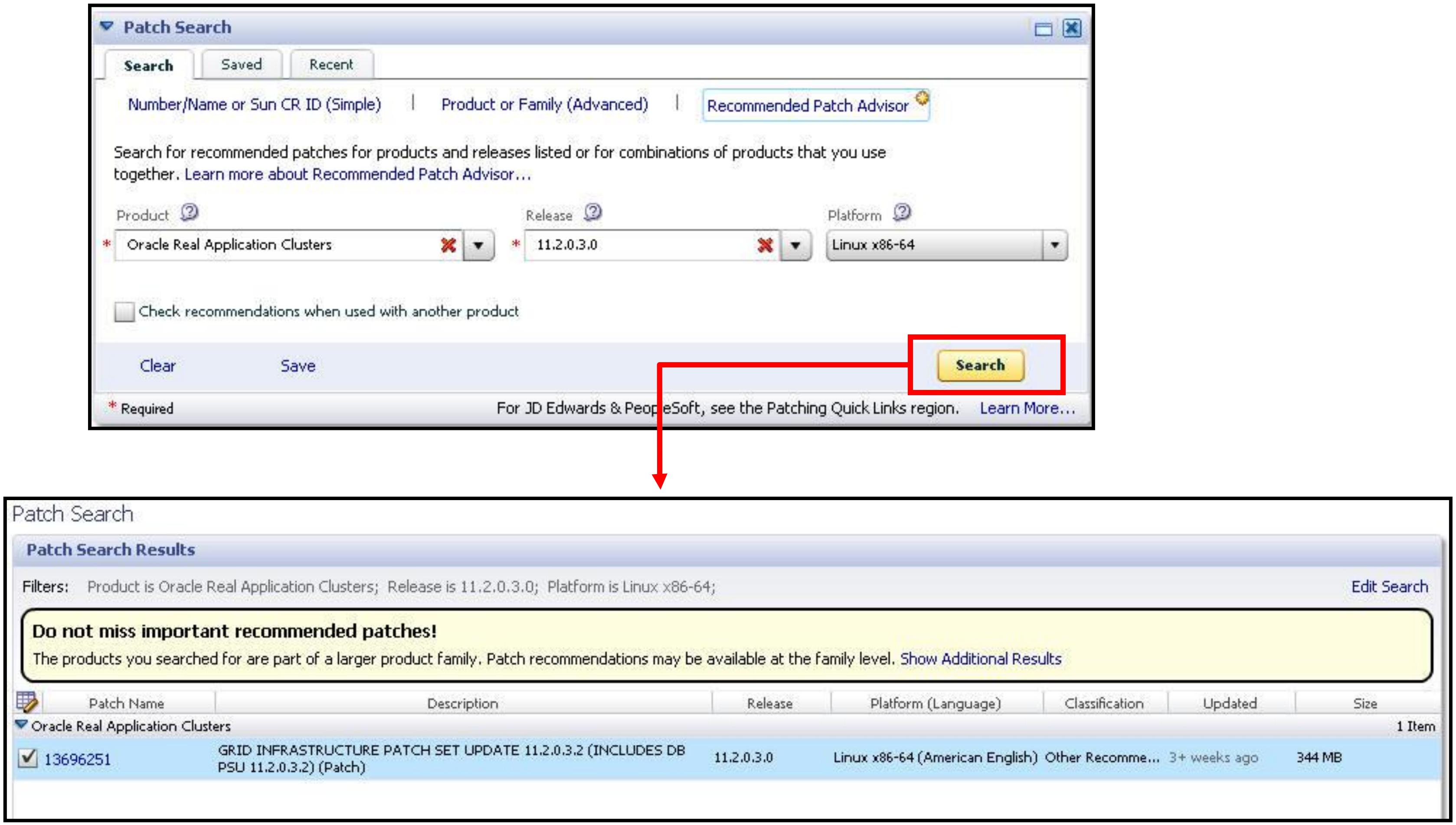


Obtaining Oracle RAC Patches

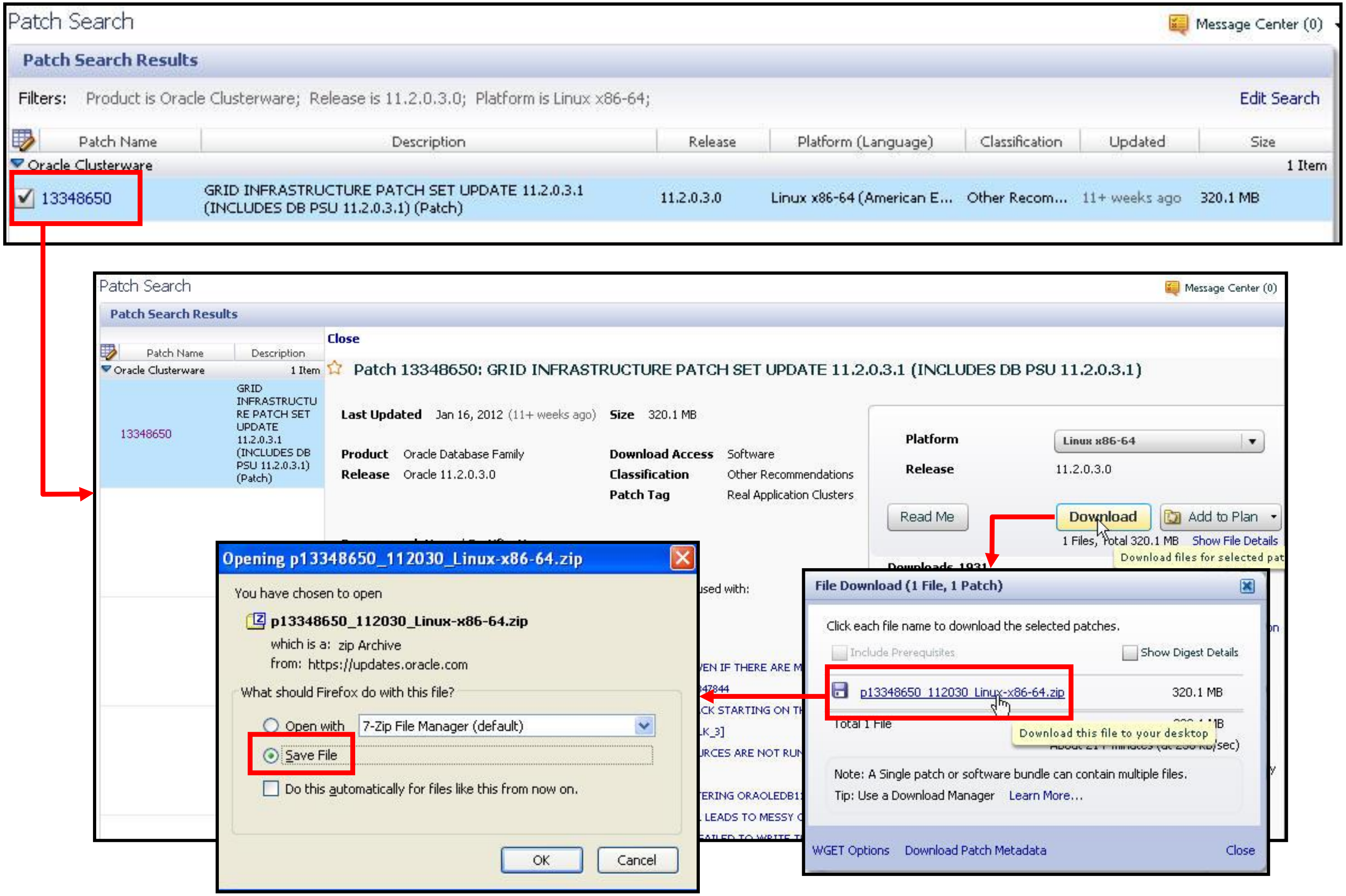


Patch Name	Description	Release	Platform (Language)	Classification	Product/Family	Updated	Size
10032375	FAST REFRESH FAILS BY ORA-1008/ORA-12801 (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	3 days ago	285.9 KB
10102373	UNABLE TO USE ORA11GR2 (11.2.0.1.0) AQAPI.JAR WHEN DOING XA FROM JCAPS 6.3 (Patch)	11.2.0.3.0	Generic Platform (American...	General	Oracle Database Fa...	6+ weeks ago	21.9 KB
10109915	ASM HANGS IN HIGH REDUNDANCY CONFIG IF 1 OF 5 DISKS GOES OFFLINE (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	4 days ago	157.3 KB
10145667	ERRORS TRYING TO REWRITE QUERY WITH EXACT TEXT MATCH TO MVIEW (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	5+ weeks ago	699.6 KB
10182005	SR12.1MAXIO - TRC - KRBRRD_COMP (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	3+ weeks ago	263.2 KB
10263668	ORA-00600 [KOKEGPINLOB1] PRINTING RESULTS OF SQL QUERY (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	3+ weeks ago	58.2 KB
10359307	KOKC LATCH RECOVERY AREA NEEDS OPTIMIZATION (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	3+ weeks ago	39.3 KB
11703330	PLACEHOLDER BUG FOR METADATA XML IN 11203 (Patch)	11.2.0.3.0	Generic Platform (American...	General	Oracle Database Fa...	4+ weeks ago	786 Byte(s)
11896575	WHEN CREATING SYNONYM FOR SYS.XMLTYPE ON DATABASE 11.2.0.2 GET ORA-22933 (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	5+ weeks ago	705.9 KB
11901407	REMOVE GATHERING STATISTICS FROM DATABASE UPGRADE PROCESS (Patch)	11.2.0.3.0	Generic Platform (American...	General	Oracle Database Fa...	1 week ago	11.3 KB
12312133	STANDBY DB CRASHES WITH ORA-600 [KRCCCB_BUSY] (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	1 week ago	636.7 KB
12358753	INDEX_STATS TABLE RETURNS WRONG VALUE (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	4+ weeks ago	122.5 KB
12378705	PLS-801 [56404] REPORTED ON ACTIVE DATA GUARD DATABASE WHEN CALLING A FUNCTION (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	1 week ago	506.2 KB
12424121	WHEN BUFFER SPLIT HAPPENED ON PI TARGET, XDKC PARSER GOT SEGMENTATION FAULT (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	6+ weeks ago	143.7 KB
12552578	ORA-01790: EXPRESSION MUST HAVE SAME DATATYPE AS CORRESPONDING EXPRESSION (Patch)	11.2.0.3.0	Linux x86-64 (American En...	General	Oracle Database Fa...	1 week ago	1.3 MB

Obtaining Oracle RAC Patches



Downloading Patches



Reduced Down-Time Patching for Cluster Environments

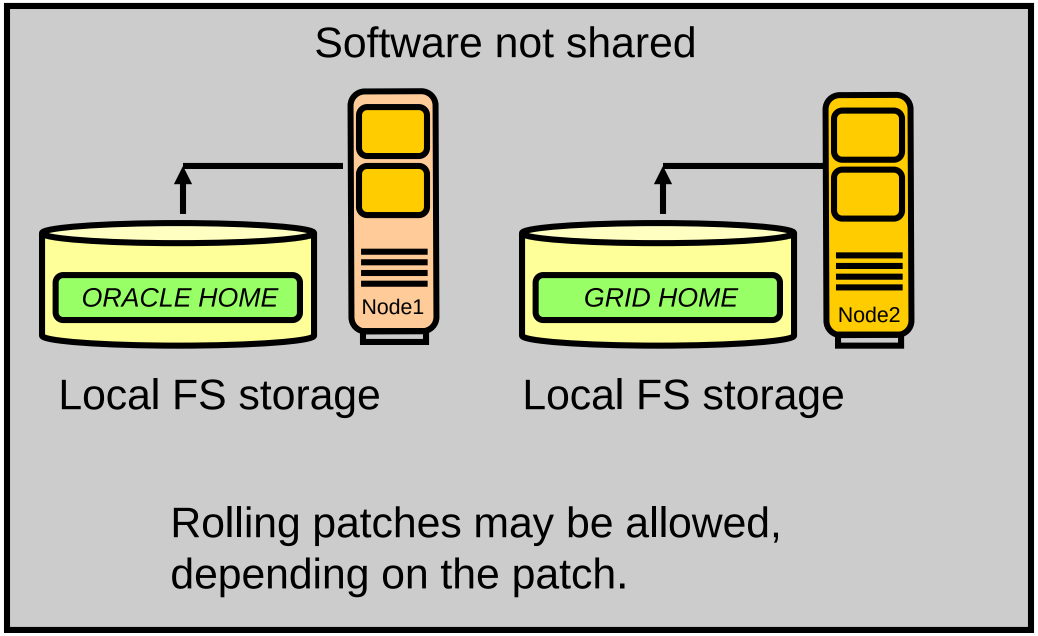
Patching Oracle RAC can be completed without taking the entire cluster down.

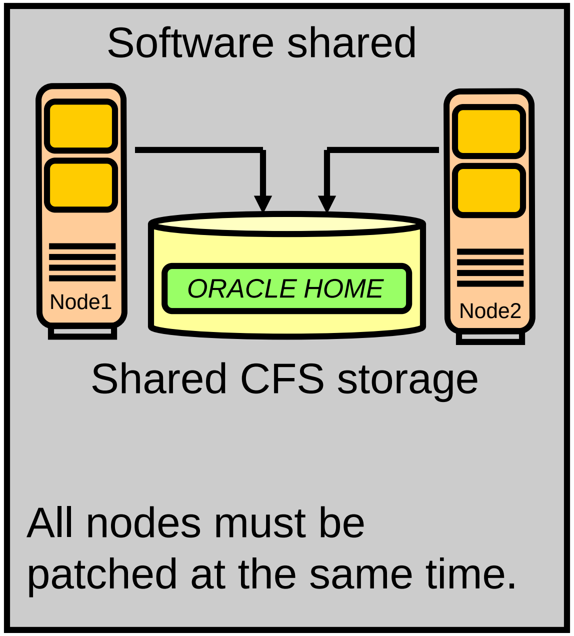
- OPatch can apply patches in multinode, multipatch fashion.
- OPatch detects whether the database schema is at an earlier patch level than the new patch, and runs SQL commands to bring the schema up to the new patch level.
- OUI installs patch sets as out-of-place upgrades, reducing the down time required for patching.

Rolling Patches

A rolling patch allows one node at a time to be patched, while other nodes continue to provide service. It:

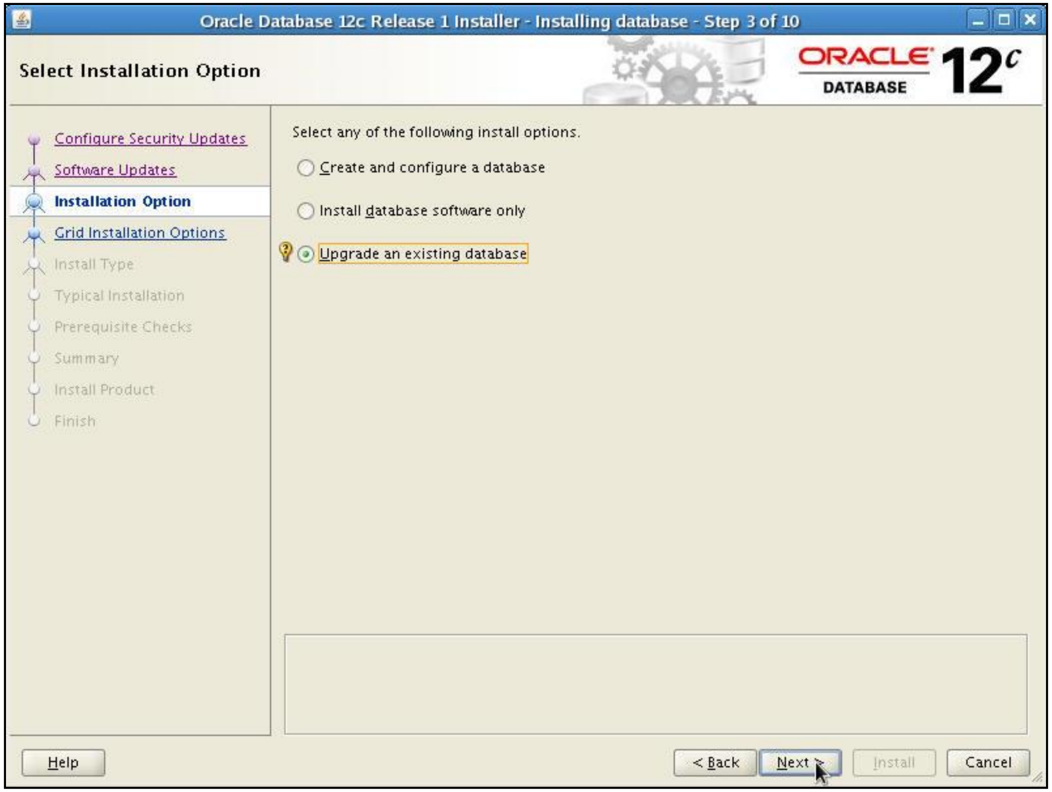
- Requires distinct software homes for each node
- Allows different versions to coexist temporarily
- May not be available for all patches





Out-of-Place Database Upgrades

- You must install the software for the new Oracle Database release before you can perform the upgrade.
- The software is installed to a new Oracle Home by using OUI.
- The Database Upgrade Assistant is used to finish the upgrade process.



Out-of-Place Database Upgrade with OUI

1. Upgrade Grid Infrastructure first, if necessary.
2. Follow the instructions in your Oracle OS-specific documentation to prepare for installation of Oracle Database software.
3. Use OUI to install the Oracle Database software.
4. Execute the Pre-Upgrade Information Tool and correct any reported deficiencies.

```
SQL> @NEW_ORACLE_HOME/rdbms/admin/preupgrd.sql
```

5. Run the `root.sh` script as directed by OUI.
6. Finish the upgrade process with DBUA.

OPatch: Overview

- OPatch is a utility that assists you with the process of applying interim patches to Oracle software.
- OPatch is a Java-based utility that allows the application and rolling back of interim patches.
- OPatch is included with the Oracle Clusterware 12*c* installation.
- For large IT environments, EM Cloud Control's patch automation capability can simplify the patching process.

OPatch: General Usage

- To define the `ORACLE_HOME` or `-oh` option on all commands:

```
$ export ORACLE_HOME=/u01/app/12.1.0/grid
$ opatch command [options]
```

or

```
$ opatch command -oh /u01/app/12.1.0/grid [options]
```

- To obtain help with the OPatch syntax:

```
$ opatch command -help
```

- To check whether a patch supports a rolling application (Run from the `patch` directory):

```
$ opatch query -all | grep -i Rolling
```

Before Patching with OPatch

- Check the current setting of the `ORACLE_HOME` variable.
- Back up the directory being patched with an OS utility or Oracle Secure backup.
- Stage the patch to each node.
- Update the `PATH` environment variable for the `OPatch` directory.

OPatch Automation

- OPatch has automated patch application for the Oracle Grid Infrastructure and Oracle RAC database homes.
- Existing configurations are queried and the steps required for patching each Oracle RAC database home of the same version and the Grid home are automated.
- The utility must be executed by an operating system user with root privileges.
- OPatch must be executed on each node in the cluster if the Grid home or RAC home is in non-shared storage.
- One invocation of OPatch can patch the Grid home, one or more RAC homes, or both Grid and Oracle RAC database homes of the same Oracle release version.

OPatch Automation: Examples

- To patch Grid home and all Oracle RAC database homes of the same version:

```
# opatchauto apply <UNZIPPED_PATCH_LOCATION> <Grid_home>
-ocmrf <ocm_response_file>
```

- To patch only the GI home:

```
# opatchauto apply <UNZIPPED_PATCH_LOCATION> -oh
<Grid_home>  -ocmrf <ocm_response_file>
```

- To patch one or more Oracle RAC database homes:

```
# opatchauto apply <UNZIPPED_PATCH_LOCATION> -database
db1, db2 -ocmrf <ocm_response_file>
```

OPatch Log and Trace Files

- OPatch maintains logs for apply, rollback, and lsinventory operations.
- OPatch Log files are located in `ORACLE_HOME/cfgtoollogs/opatch`
- Each log file is tagged with the time stamp of the operation.
- Each time you run OPatch, a new log file is created.
- OPatch maintains an index of processed commands and log files in the `opatch_history.txt` file.

Queryable Patch Inventory

- The DBMS_QOPATCH package provides a PL/SQL or SQL interface to view the database patches that are installed.
- The interface provides all the patch information available as part of the OPatch lsinventory -xml command.
- The package accesses the OUI patch inventory in real time to provide patch and patch meta information.
- The DBMS_QOPATCH package allows users to:
  - Query what patches are installed from SQL*Plus
  - Write wrapper programs to create reports and do validation checks across multiple environments
  - Check patches installed on cluster nodes from a single location

Quiz

Which tools can be used to install a patch set?

a. Oracle Universal Installer
b. OPatch
c. Enterprise Manager Database Console
d. Database Configuration Assistant

Summary

In this lesson, you should have learned how to:

- Describe the types of patches available
- Plan for rolling patches and rolling upgrades
- Install a patch set with the Oracle Universal Installer (OUI) utility
- Install a patch with the `opatch` utility

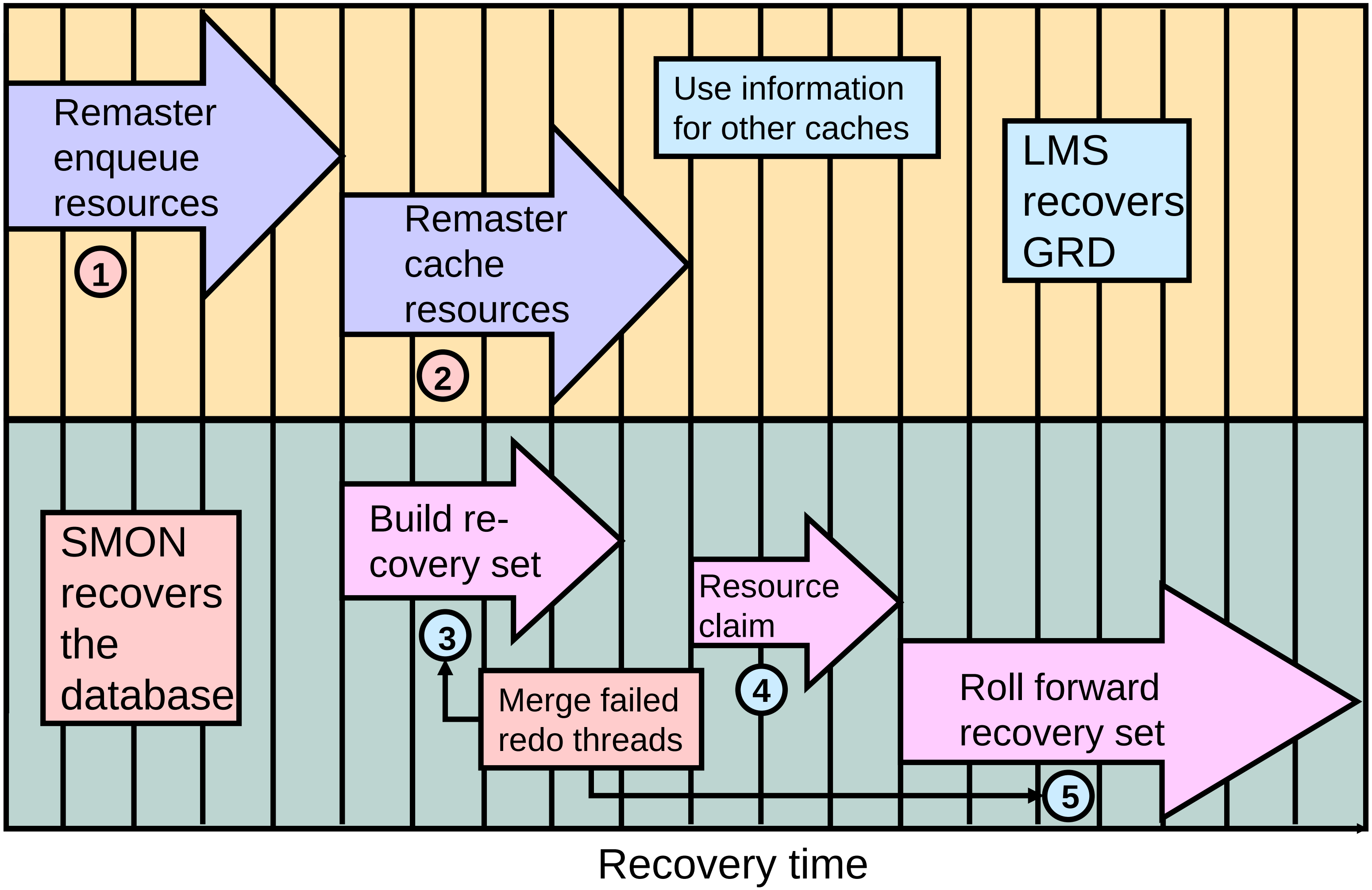
Managing Backup and Recovery for RAC

Objectives

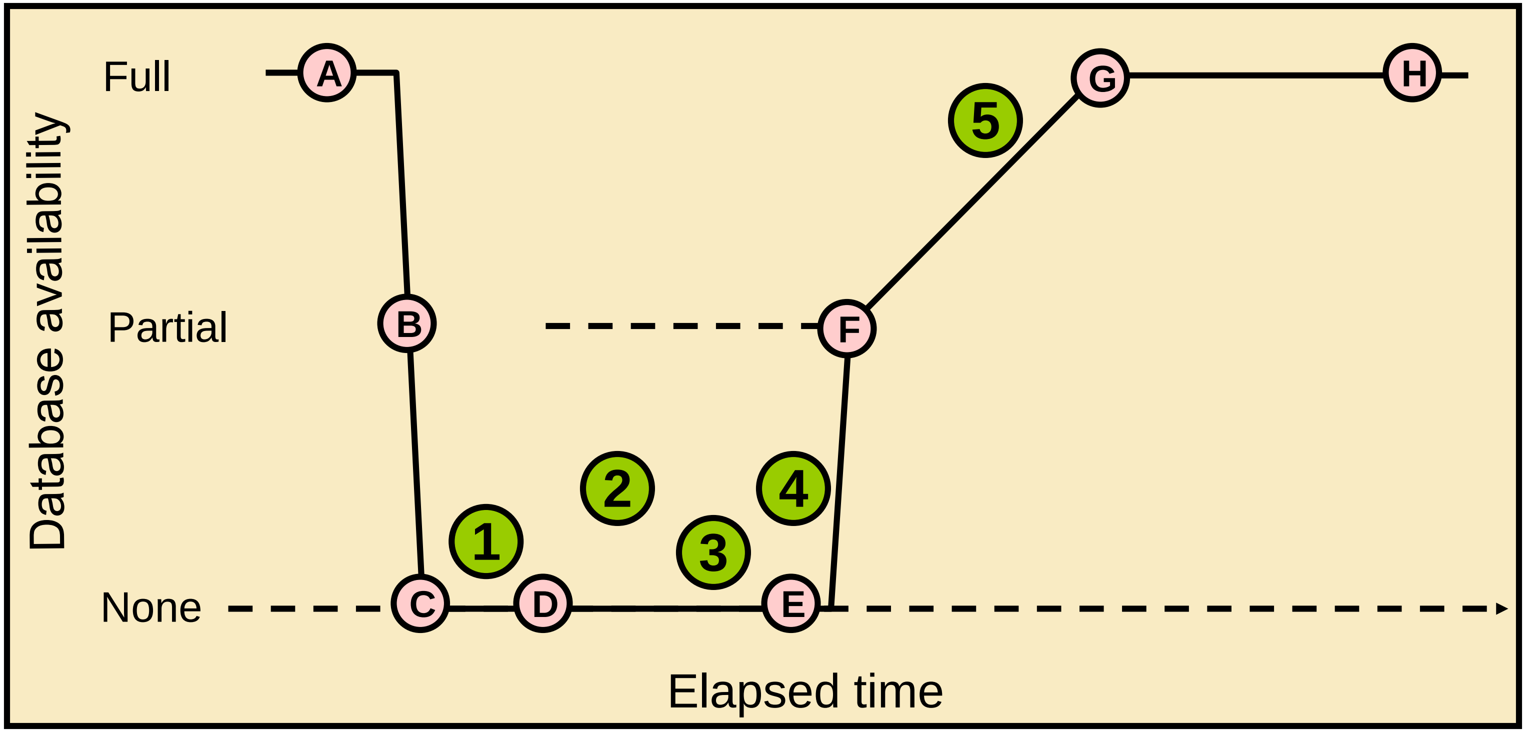
After completing this lesson, you should be able to configure the following:

- The RAC database to use `ARCHIVELOG` mode and the fast recovery area
- RMAN for the RAC environment

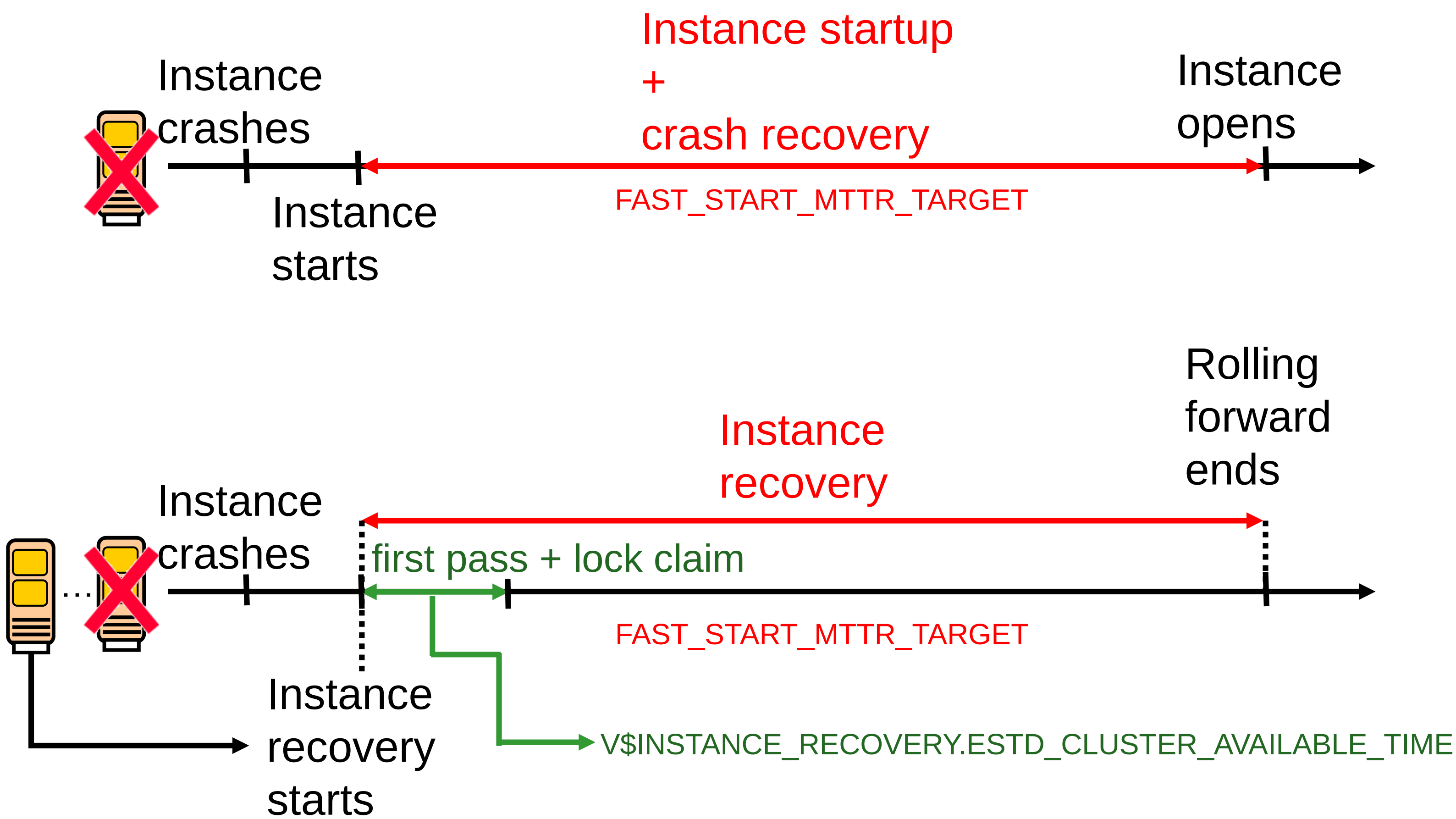
RAC and Instance Recovery



Instance Recovery and Database Availability



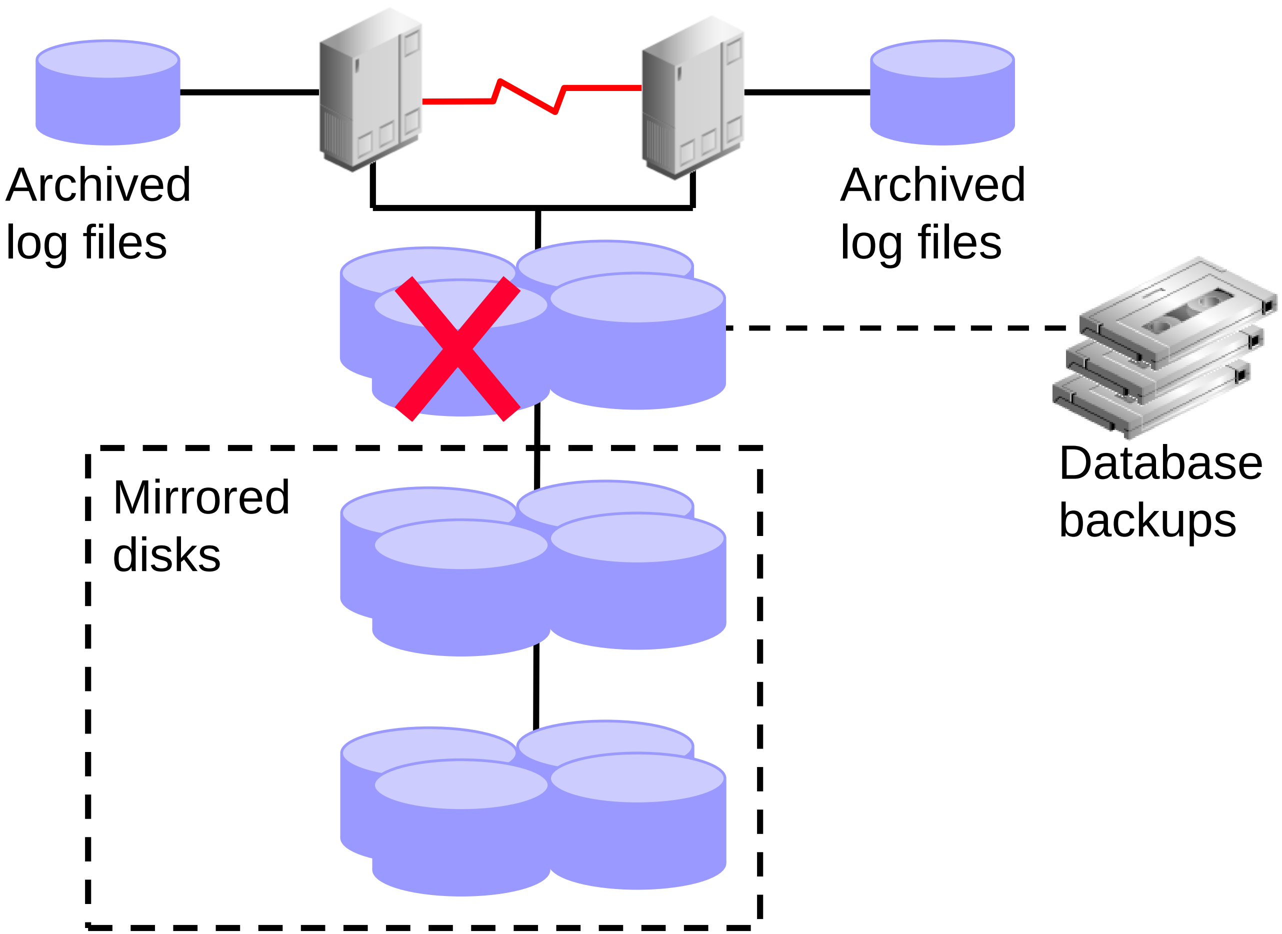
Instance Recovery and RAC



Instance Recovery and RAC

- Use parallel instance recovery.
- Set PARALLEL_MIN_SERVERS.
- Use asynchronous input/output (I/O).
- Increase the size of the default buffer cache.

Protecting Against Media Failure



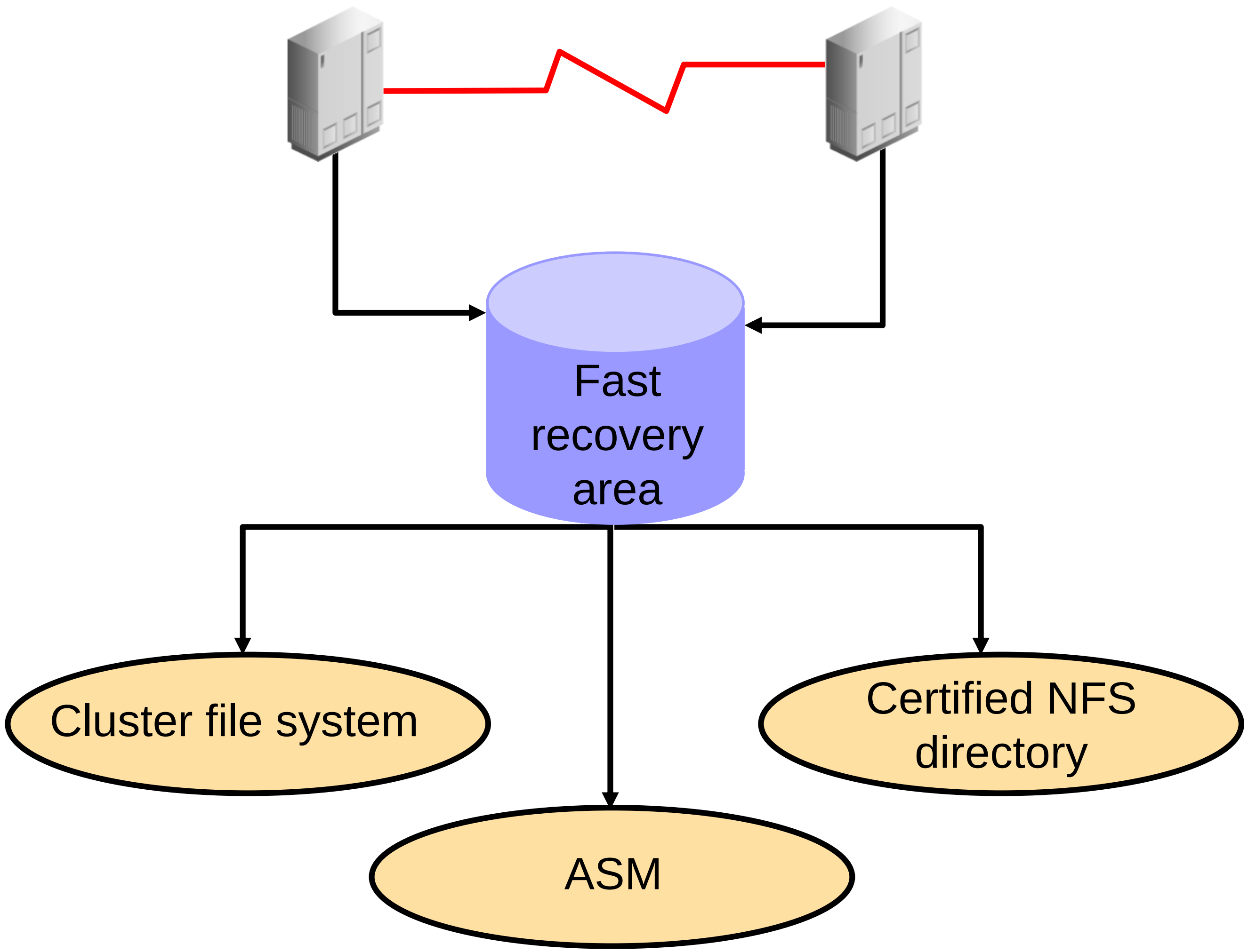
Media Recovery in Oracle RAC

- Media recovery must be user-initiated through a client application.
- In these situations, use RMAN to restore backups of the data files and then recover the database.
- RMAN media recovery procedures for RAC do not differ substantially from those for single-instance environments.
- The node that performs the recovery must be able to restore all of the required data files.
  - That node must also be able to either read all required archived redo logs on disk or restore them from backups.
- When recovering a database with encrypted tablespaces, the wallet must be opened after database mount and before you open the database.

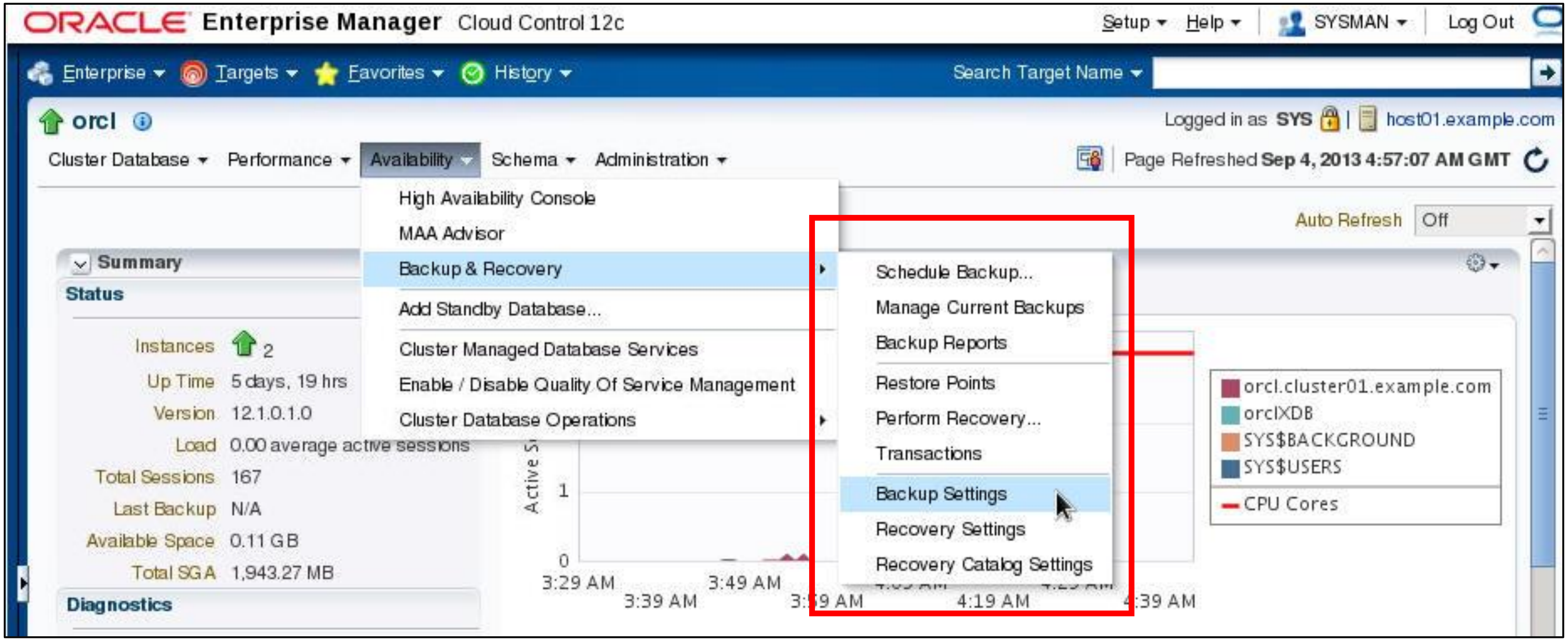
Parallel Recovery in RAC

- Oracle Database automatically selects the optimum degree of parallelism for:
  - Instance recovery
  - Crash recovery
- Archived redo logs are applied using an optimal number of parallel processes based on the availability of CPUs.
- With RMAN's `RESTORE` and `RECOVER` commands, the following three stages of recovery can use parallelism:
  - Restoring data files
  - Applying incremental backups
  - Applying archived redo logs
- To disable parallel instance and crash recovery, set the `RECOVERY_PARALLELISM` parameter to 0.

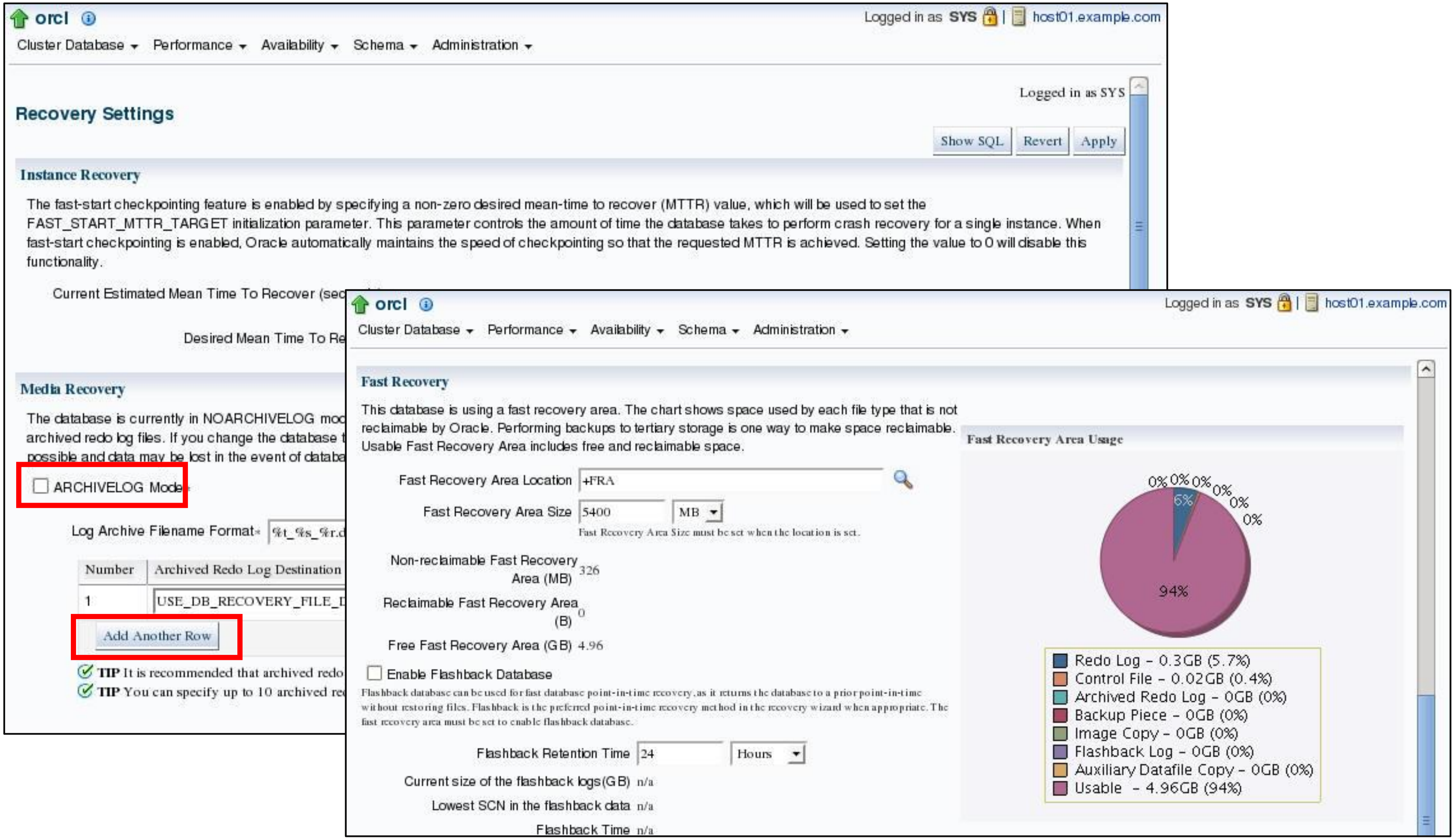
RAC and the Fast Recovery Area



RAC Backup and Recovery Using EM



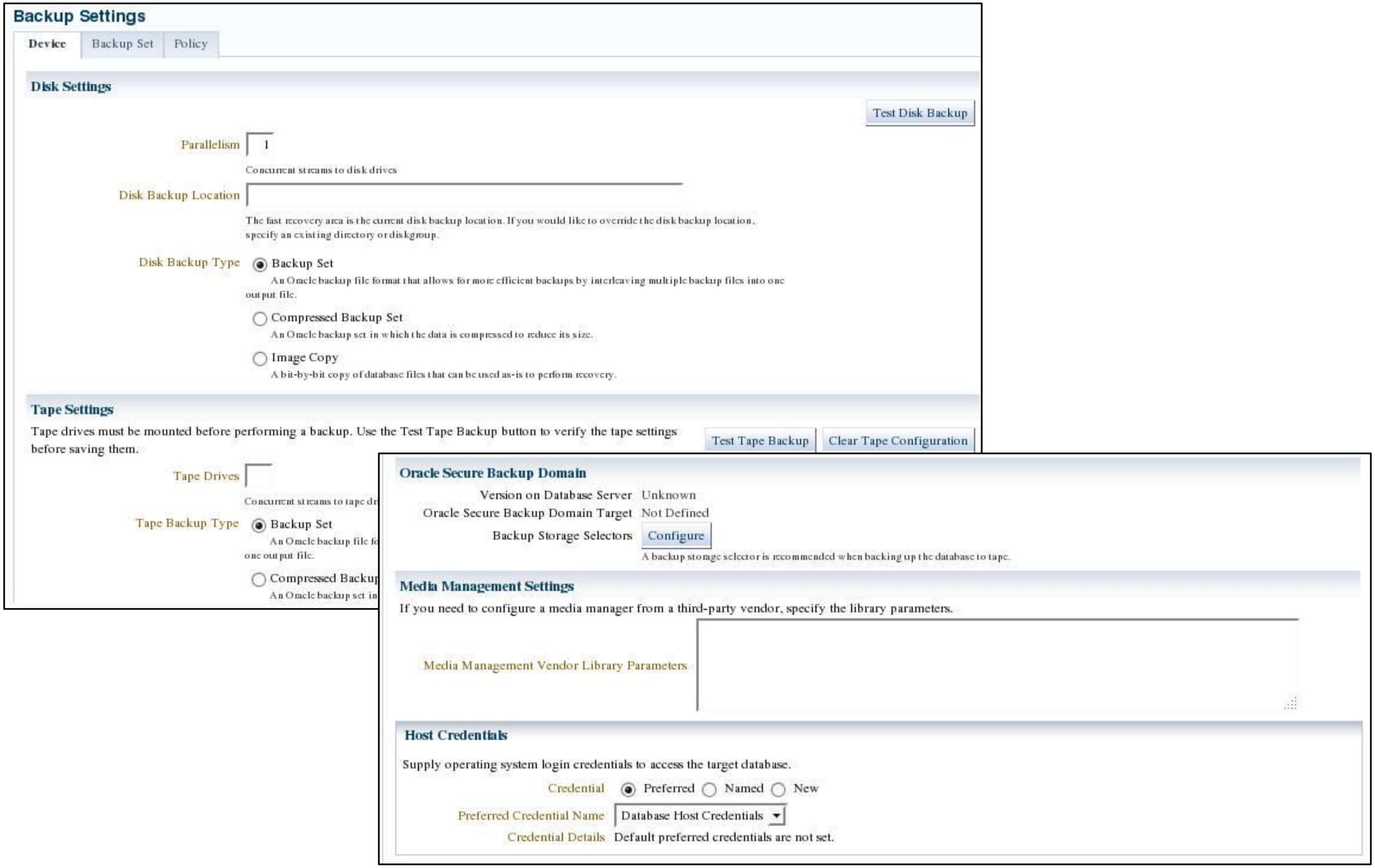
Configuring RAC Recovery Settings with EM



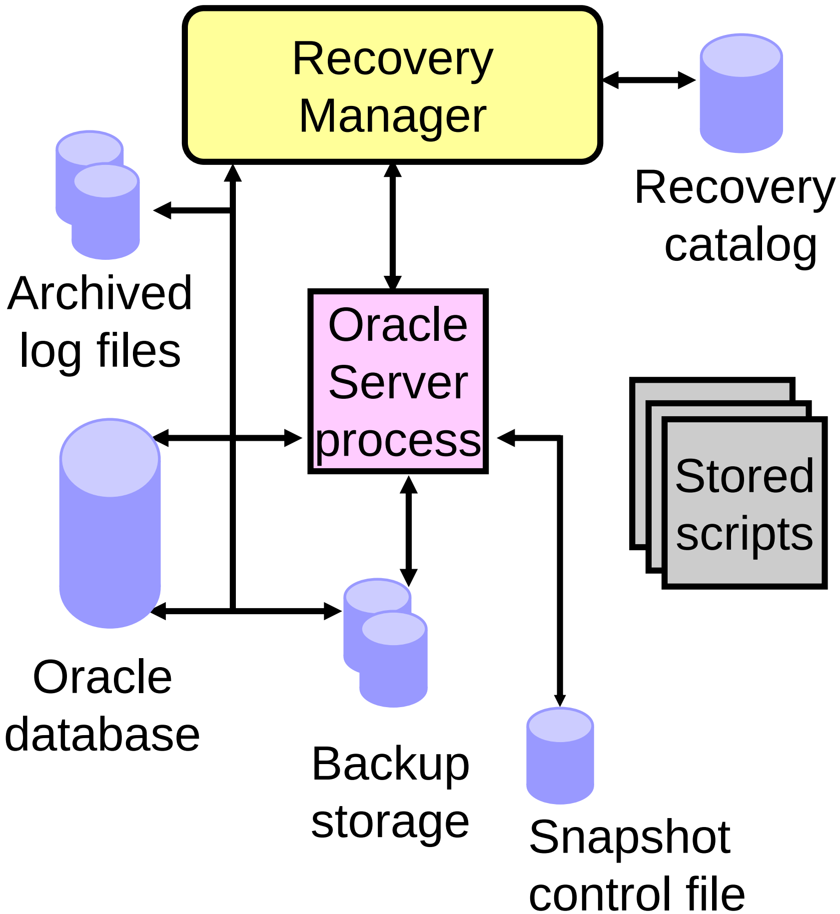
Archived Redo File Conventions in RAC

Variable	Description	Example
`%t`	Thread number, not padded	`log_1`
`%T`	Thread number, left-zero-padded	`log_0001`
`%s`	Log sequence number, not padded	`log_251`
`%S`	Log sequence number, left-zero-padded	`log_0000000251`
`%r`	Resetlogs identifier	`log_23452345`
`%R`	Padded resetlogs identifier	`log_0023452345`
`%t_%s_%r`	Using multiple variables	`log_1_251_23452345`

Configuring RAC Backup Settings with EM



Oracle Recovery Manager



RMAN provides the following benefits for Real Application Clusters:

- Can read cluster files or ASM files with no configuration changes
- Can access multiple archive log destinations

Configuring RMAN Snapshot Control File Location

- The snapshot control file path ***must*** be valid on every node from which you might initiate an RMAN backup.
- Configure the snapshot control file location in RMAN.
  - Determine the current location:

```
RMAN> SHOW SNAPSHOT CONTROLFILE NAME;
'/u01/app/oracle/product/12.1.0/dbhome_1/dbs/snapcf_orcl_3.f'
```

  - You can use ASM or a shared file system location:

```
RMAN> CONFIGURE SNAPSHOT CONTROLFILE NAME TO
'+FRA/SNAP/snap_prod.cf';
RMAN> CONFIGURE SNAPSHOT CONTROLFILE NAME TO
'/ocfs2/oradata/dbs/scf/snap_prod.cf';
```

Configuring Control File and SPFILE Autobackup

- RMAN automatically creates a control file and SPFILE backup after the `BACKUP` or `COPY` command:

```
RMAN> CONFIGURE CONTROLFILE AUTOBACKUP ON;
```

- Change default location:

```
RMAN> CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE
TYPE DISK TO '+DATA';
```

- Location ***must*** be available to all nodes in your RAC database.

Crosschecking on Multiple RAC Clusters Nodes

When crosschecking on multiple nodes, make sure that all backups can be accessed by every node in the cluster.

- This allows you to allocate channels at any node in the cluster during restore or crosscheck operations.
- Otherwise, you must allocate channels on multiple nodes by providing the `CONNECT` option to the `CONFIGURE CHANNEL` command.
- If backups are not accessible because no channel was configured on the node that can access those backups, then those backups are marked `EXPIRED`.

Channel Connections to Cluster Instances

- When backing up, each allocated channel can connect to a different instance in the cluster.
- Instances to which the channels connect must be either all mounted or all open.
- When choosing a channel to use, RMAN gives preference to the nodes with faster access to the data files that you want to back up.

```
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE CHANNEL 1 DEVICE TYPE sbt CONNECT='sys/rac@orcl_1';
CONFIGURE CHANNEL 2 DEVICE TYPE sbt CONNECT='sys/rac@orcl_2';
CONFIGURE CHANNEL 3 DEVICE TYPE sbt CONNECT='sys/rac@orcl_3';
```

OR

```
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE CHANNEL DEVICE TYPE sbt CONNECT='sys/rac@bkp_serv';
```

RMAN Channel Support for the Grid

- RAC allows the use of nondeterministic connect strings.
- RMAN can use connect strings that are not bound to a specific instance in the Grid environment.
- It simplifies the use of parallelism with RMAN in a RAC environment.
- It uses the load-balancing characteristics of the Grid environment.
  - Channels connect to RAC instances that are the least loaded.

```
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
```

RMAN Default Autolocation

- Recovery Manager autolocates the following files:
  - Backup pieces
  - Archived redo logs during backup
  - Data file or control file copies
- If local archiving is used, a node can read only those archived logs that were generated on that node.
- When restoring, a channel connected to a specific node restores only those files that were backed up to the node.

Distribution of Backups

Several possible backup configurations for RAC:

- A dedicated backup server performs and manages backups for the cluster and the cluster database.
- One node has access to a local backup appliance and performs and manages backups for the cluster database.
- Each node has access to a local backup appliance and can write to its own local backup media.

Managing Archived Redo Logs Using RMAN

- When an archived redo log is generated, Oracle records the name of the log in the control file of the target database.
- If a recovery catalog is used, RMAN records the archived redo log names in the catalog when a resynch occurs.
- The backup and recovery strategy that you choose depends on how the archive destinations are configured.

Noncluster File System Local Archiving Scheme

- When archiving locally to a noncluster file system, each node archives to a uniquely named local directory.
- If recovery is required, configure the recovery node so that it can access directories on the other nodes remotely.
  - Use NFS on Linux and UNIX computers.
  - Use mapped drives on Windows systems.
- Each node writes to a local destination, but can read archived redo log files in remote directories on other nodes.
- If noncluster local archiving is used for media recovery, configure the node performing recovery for remote access.

Configuring Non-Cluster, Local Archiving

- Set the `SID.LOG_ARCH_DEST` parameter for each instance using the SID designator:

```
sid1.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_1"
sid2.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_2"
sid3.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_3"
```

- For policy-managed databases, create a node and instance binding to ensure that sid1 always runs on the same node:

```
$ srvctl modify instance -d mydb -n node1 -i sid1
$ srvctl modify instance -d mydb -n node2 -i sid2
$ srvctl modify instance -d mydb -n node3 -i sid3
```

ASM and Cluster File System Archiving Scheme

- The preferred configuration for RAC is to use ASM disk group for a recovery area for the recovery set that is different from the disk group used for data files.
  - When you use ASM, it uses an OMF naming format.
- Alternatively, a cluster file system archiving scheme can be used.
  - Each node writes to a single location on the cluster file system when archiving the redo log files.
  - Each node can read the archived redo log files of the other nodes.
- The advantage of this scheme is that none of the nodes uses the network to archive logs.

Configuring the CFS Archiving Scheme

- In the CFS scheme, each node archives to a directory with the same name on all instances within the database.
- To configure this directory, set values for the `LOG_ARCH_DEST_1` parameter:

```
*.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
```

- The below list shows archived redo log entry examples that would appear in the RMAN catalog or in the control file:

```
/arc_dest/log_1_999_23452345.arc
/arc_dest/log_1_1000_23435343.arc
/arc_dest/log_1_1001_23452345.arc <- thread 1 archived in node3
/arc_dest/log_3_1563_23452345.arc <- thread 3 archived in node2
/arc_dest/log_2_753_23452345.arc <- thread 2 archived in node1
/arc_dest/log_2_754_23452345.arc
```

- Each node can read the logs written by itself and any other node.

Restoring and Recovering

- Media recovery may require one or more archived log files from each thread.
- The `RMAN RECOVER` command automatically restores and applies the required archived logs.
- Archive logs may be restored to any node performing the restore and recover operation.
- Logs must be readable from the node performing the restore and recovery activity.
- Recovery processes request additional threads enabled during the recovery period.
- Recovery processes notify you of threads no longer needed because they were disabled.

Quiz

Which of the following statements regarding media recovery in RAC is *not* true?

a. Media recovery must be user-initiated through a client application.

b. RMAN media recovery procedures for RAC are quite different from those for single-instance environments.

c. The node that performs the recovery must be able to restore all the required data files.

d. The recovering node must be able to either read all required archived redo logs on disk or restore them from backups.

Quiz

To use a fast recovery area in RAC, you must place it on an ASM disk group, a cluster file system, or on a shared directory that is configured through certified NFS for each RAC instance.

a. True

b. False

Summary

In this lesson, you should have learned how to configure the following:

- The RAC database to use `ARCHIVELOG` mode and the fast recovery area
- RMAN for the RAC environment

Practice 6: Overview

This practice covers the following topics:

- Configuring the archive log mode
- Configuring specific instance connection strings
- Configuring RMAN and performing parallel backups

Global Resource Management Concepts

Objectives

After completing this lesson, you should be able to describe:

- The need for global concurrency control
- Global Resource Directory
- How global resources are managed
- RAC global resource access coordination
  - Global enqueue and instance lock management
  - Global buffer cache management

Need for Global Concurrency Control

- Oracle requires concurrency control because it is a multi-user system.
- Single-instance Oracle provides concurrency control:
  - Latches or mutexes for memory structures
  - Enqueues for resource control
  - Buffer cache pins for cache management
- In RAC, structures and resources may be accessed by or modified by a session running on any database instance.
- RAC, therefore, requires additional global concurrency controls to mediate access across instances.
  - Global locks control library and row cache access.
  - Global enqueues control resource access.
  - Cache fusion controls buffer cache access.

Global Resource Directory (GRD)

- An object under global concurrency control is called a resource.
- Resource metadata is held in the Global Resource Directory (GRD).
  - Global enqueue resources are used for enqueues and locks.
  - Global cache resources are used for buffer cache control.
- The GRD is distributed among all active instances of each database or ASM environment.
- Each currently managed GRD resource has:
  - A master metadata structure
  - One or more shadow metadata structures
- The GRD uses memory from the shared pool.

Global Resource Management

- After first access of a globally managed entity by any instance, a global resource is allocated.
- An internal algorithm is used to decide which instance should contain the master metadata structure for that entity.
  - This instance is known as the resource master.
  - The resource mastering instance may be any active instance of the database or ASM environment.
- Subsequent access to an entity from another instance for which resource master metadata exists causes resource shadow metadata to be allocated in the requesting instance and updates to be done to the master metadata.

Global Resource Remastering

- Remastering is the process of allocating control of the master metadata for a specific entity to another instance.
- Instance-level or lazy remastering occurs when:
  - A new instance of the same database or ASM starts
  - A current instance is shut down gracefully
- File affinity remastering occurs when:
  - Requests to access blocks in a data file occur frequently from an instance, and the resource masters for the blocks are often held by other instances
- Object-affinity remastering occurs when:
  - Requests to access blocks in a data object occur frequently from an instance, and the resource masters for the blocks are often held by other instances

Global Resource Recovery

- When one or more but not all instances fail:
  - The failing instance(s) resource masters are lost
  - Any resource master that had a shadow in a surviving instance must be recovered
- The surviving instances can rebuild resource master metadata for a specific resource, by aggregating details from surviving shadow metadata for the same resource.
- Global locks and enqueue metadata are done first, followed by global cache metadata.
- The rebuilding results in each surviving instance mastering some of the recovered resource master metadata.

Global Resource Background Processes

- **`ACMS`**: Atomic Control file to Memory Service
- **`LMHB`**: Monitors `LMON`, `LMD`, and `LMS`*n* processes
- **`LMD0`**: Requests global enqueues and instance locks
- **`LMON`**: Issues heartbeats and performs recovery
- **`LMS`*n***: Processes global cache fusion requests
- **`LCK0`**: Is involved in library and row cache locking
- **`RCBG`**: Processes Global Result Cache invalidations

Global Resource Access Coordination

- There are two types of global resource coordination.
- Global enqueue management, which is used for:
  - Global enqueues
  - Global instance locks
- Global buffer cache, which:
  - Is also known as cache fusion or global cache
  - Is a logical buffer cache spanning all instances
  - Coordinates access to block images in the global cache
  - Supports Parallel Query across the global cache

Global Enqueues

Processing starts in the requesting instance as follows:

1. A global enqueue request is made by a session.
2. The request is passed to `LMD0` in the requesting instance.
3. The foreground waits for the request on event.
4. `LMD0` determines the mastering instance.
5. `LMD0` forwards the request to the mastering instance if required.
6. The mastering instance adds a new master resource if required.
   – Process is made an owner, waiter, or converter as appropriate.
   – Once the resource can be granted to the requestor, `LMD0` in the mastering instance notifies `LMD0` in the requesting instance.
7. When the resource is available, the foreground is posted by `LMD0` in the requesting instance.

Instance Locks

- Instance locks are used to represent which instance(s) has (have) control over an instance-wide structure:
  - Row cache entries
  - Library cache entries
  - Result cache entries
- The owner, waiter, or converter on an instance lock is the `LCK0` process.
  - As long as the local `LCK0` process in an instance owns the lock on a specific resource, any session in that instance can use the cached metadata, because it is considered current.
  - If the local instance does not own the lock, then a request must be made for the lock and the foreground waits on `DFS Lock Handle` wait event.

Global Cache Management: Overview

Global cache management provides:

- A concurrency mechanism for multiple buffer caches
- An optimization of block access for reads
- An optimization of writes for dirty buffers
- A mechanism to optimize parallel queries

Global Cache Management Components

- The `LMSn` processes
- Buffers
- Buffer headers
- Global Cache Master Resources
- Global Cache Shadow Resources

Global Cache Buffer States

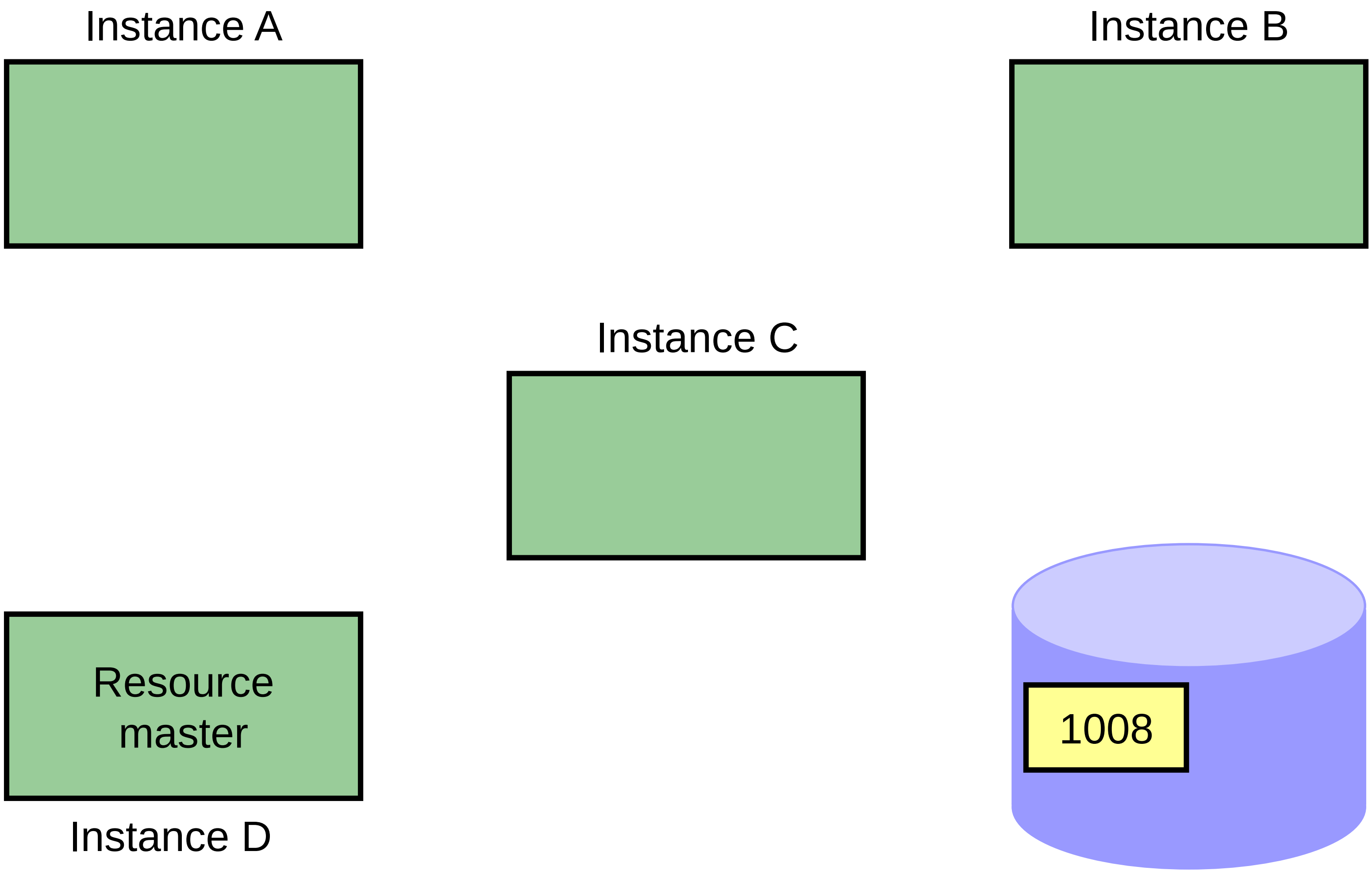
- Buffer states are visible in `V$BH.STATUS`.
- Important buffer states are:
  - Shared Current: `SCUR`
  - Exclusive Current: `XCUR`
  - Consistent Read: `CR`
    - Built in the Instance
    - Sent by cache fusion
    - Converted from `SCUR` or `PI`
  - Past Image: `PI`
    - Converted from `XCUR`
    - Not normally written
    - Converted to `CR` after later `XCUR` image is written
    - Multiple PI images may exist for same block in different buffer caches.

Global Cache Management Scenarios for Single Block Reads

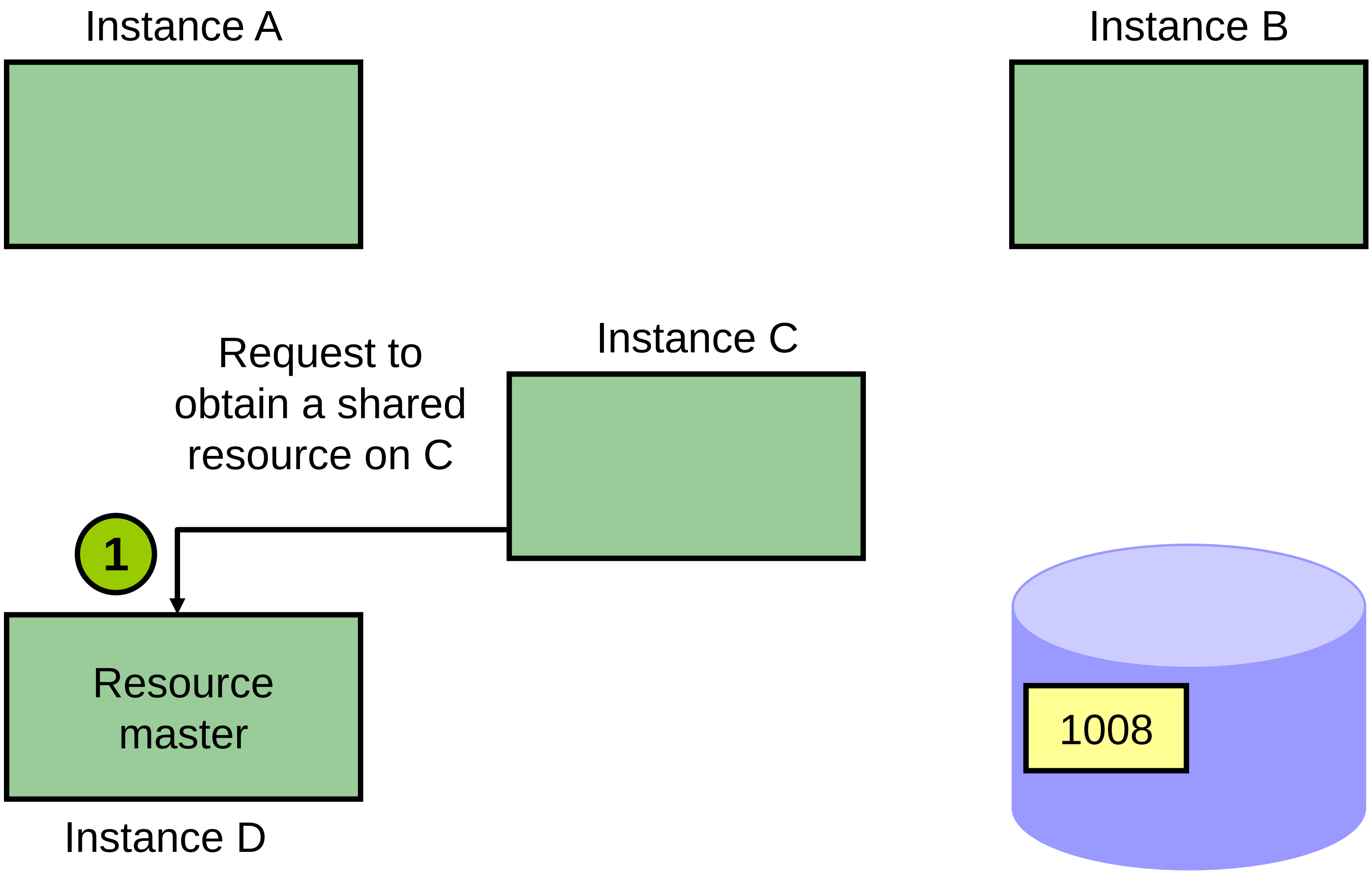
There are several scenarios for single block reads:

- Read from Disk
- Read – Read
- Read – Write
- Write – Write
- Write – Read
- Write to Disk

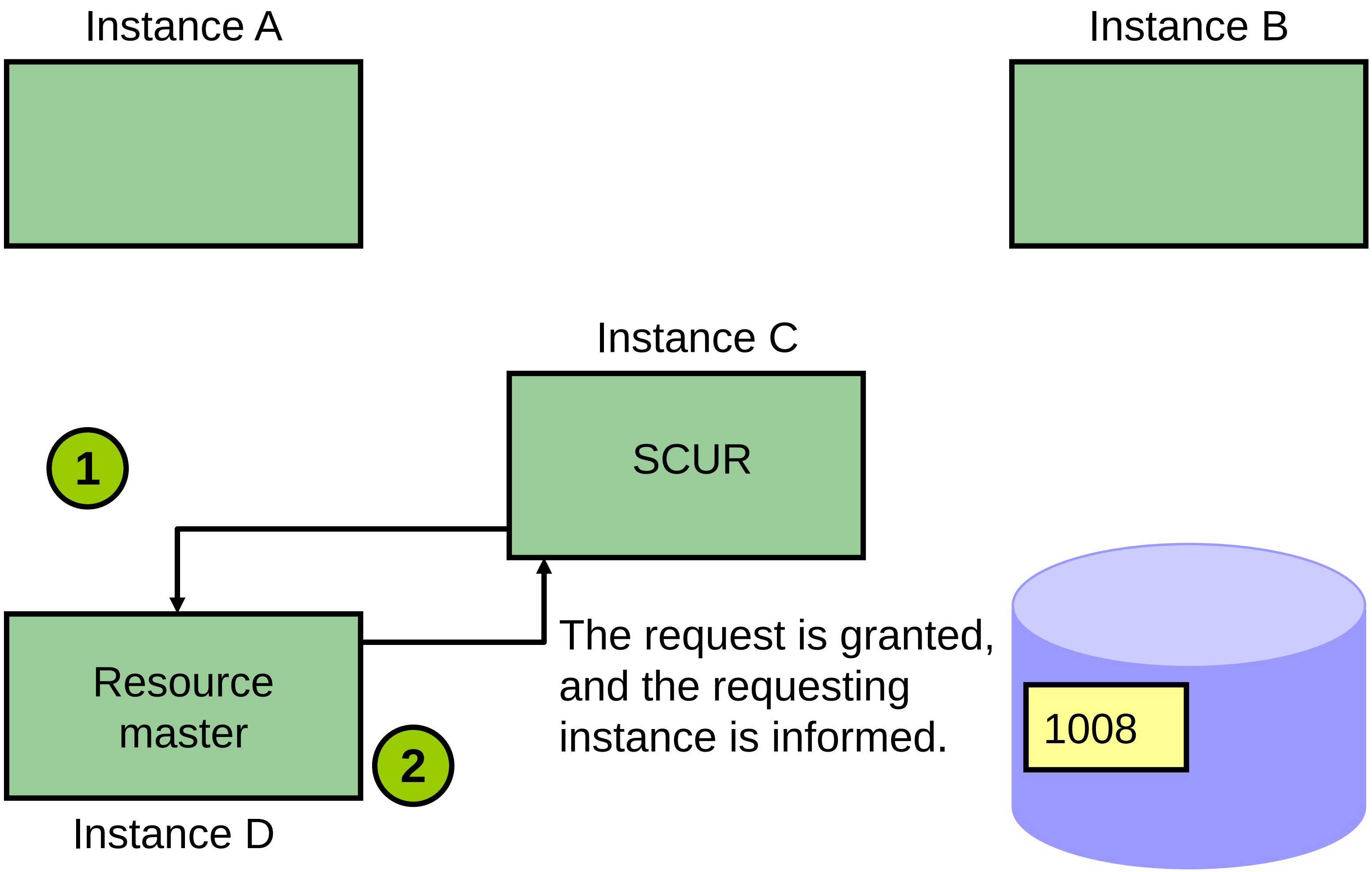
Global Cache Scenarios: Overview



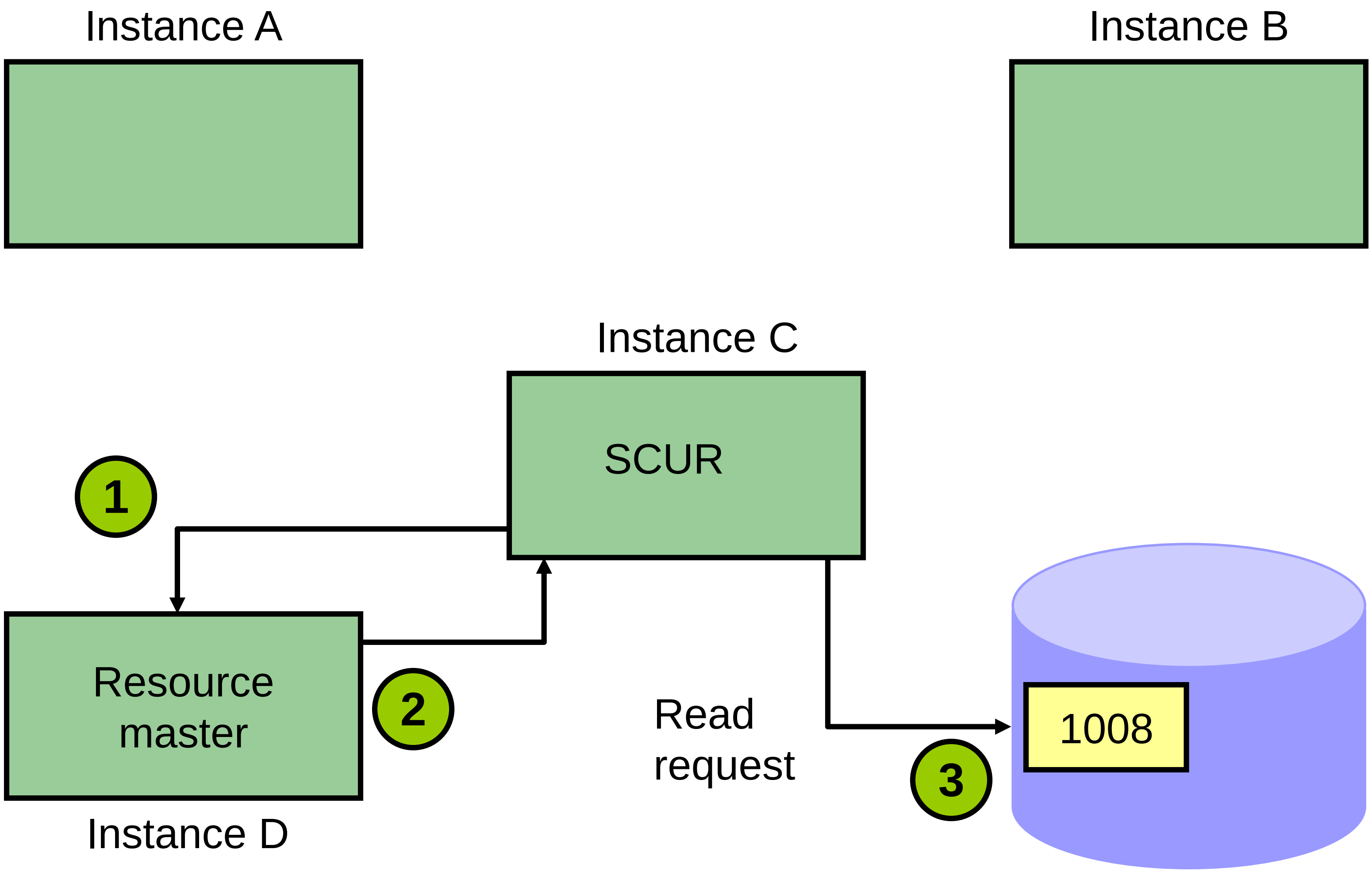
Scenario 1: Read from Disk



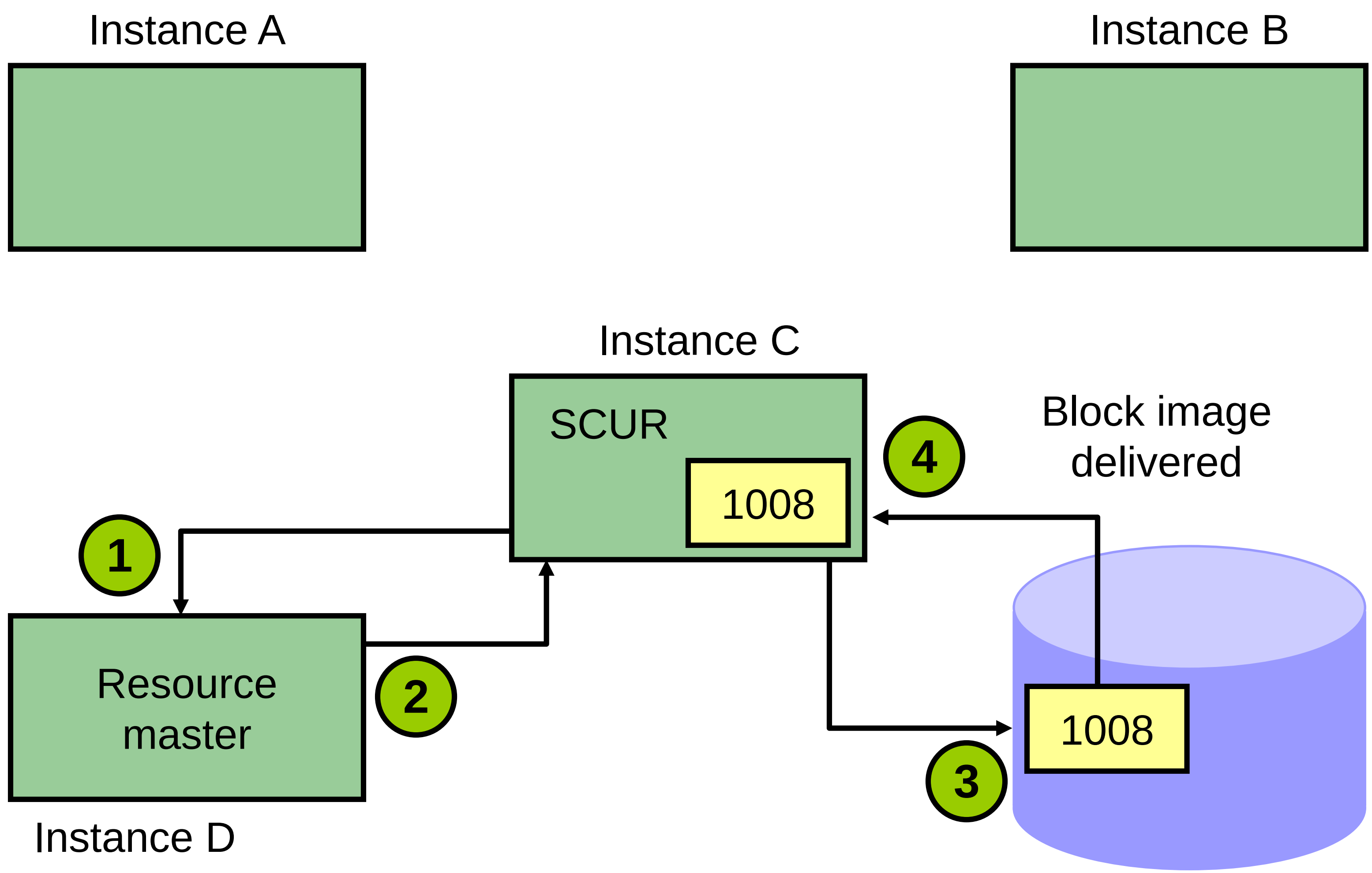
Scenario 1: Read from Disk



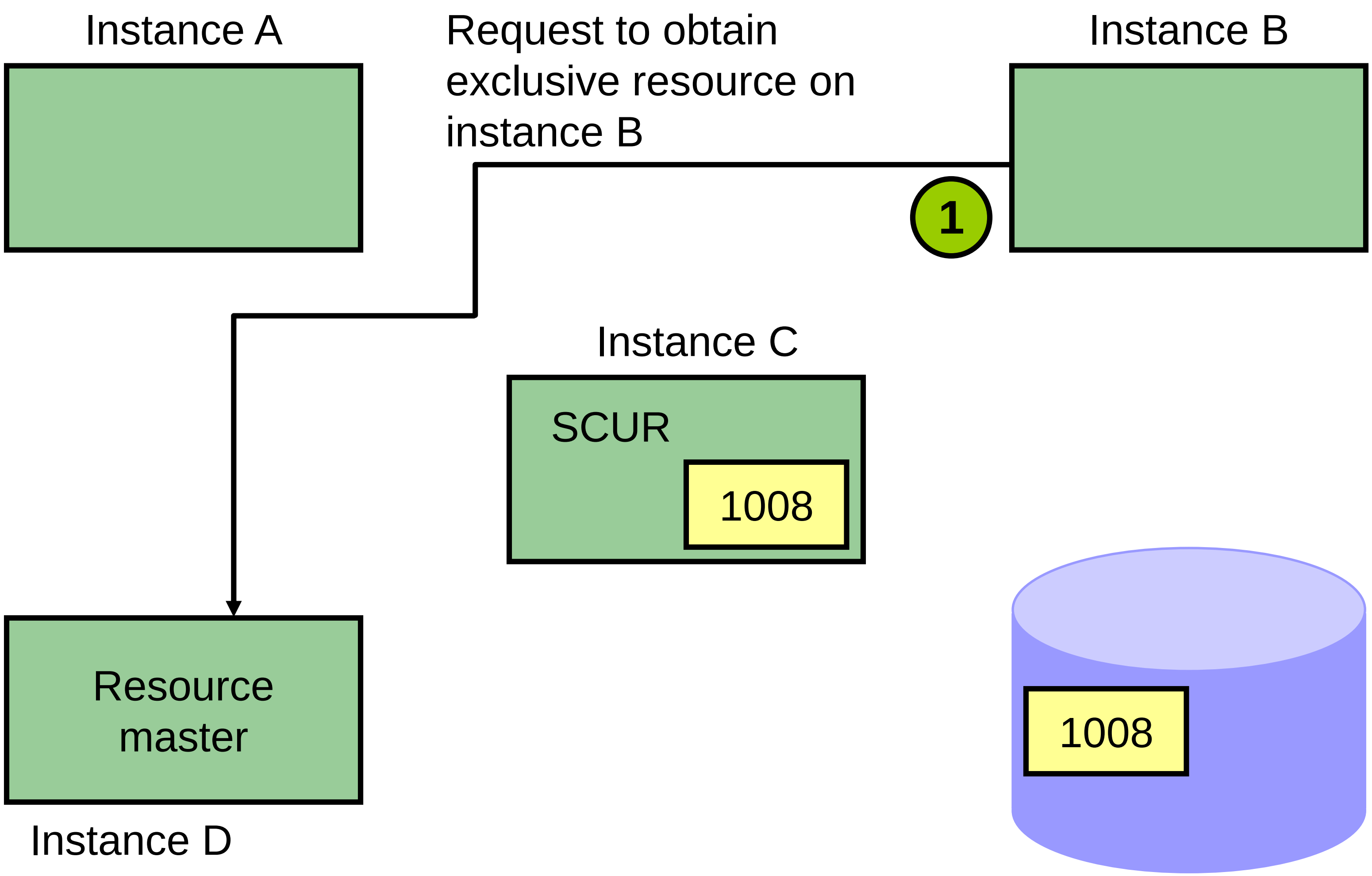
Scenario 1: Read from Disk



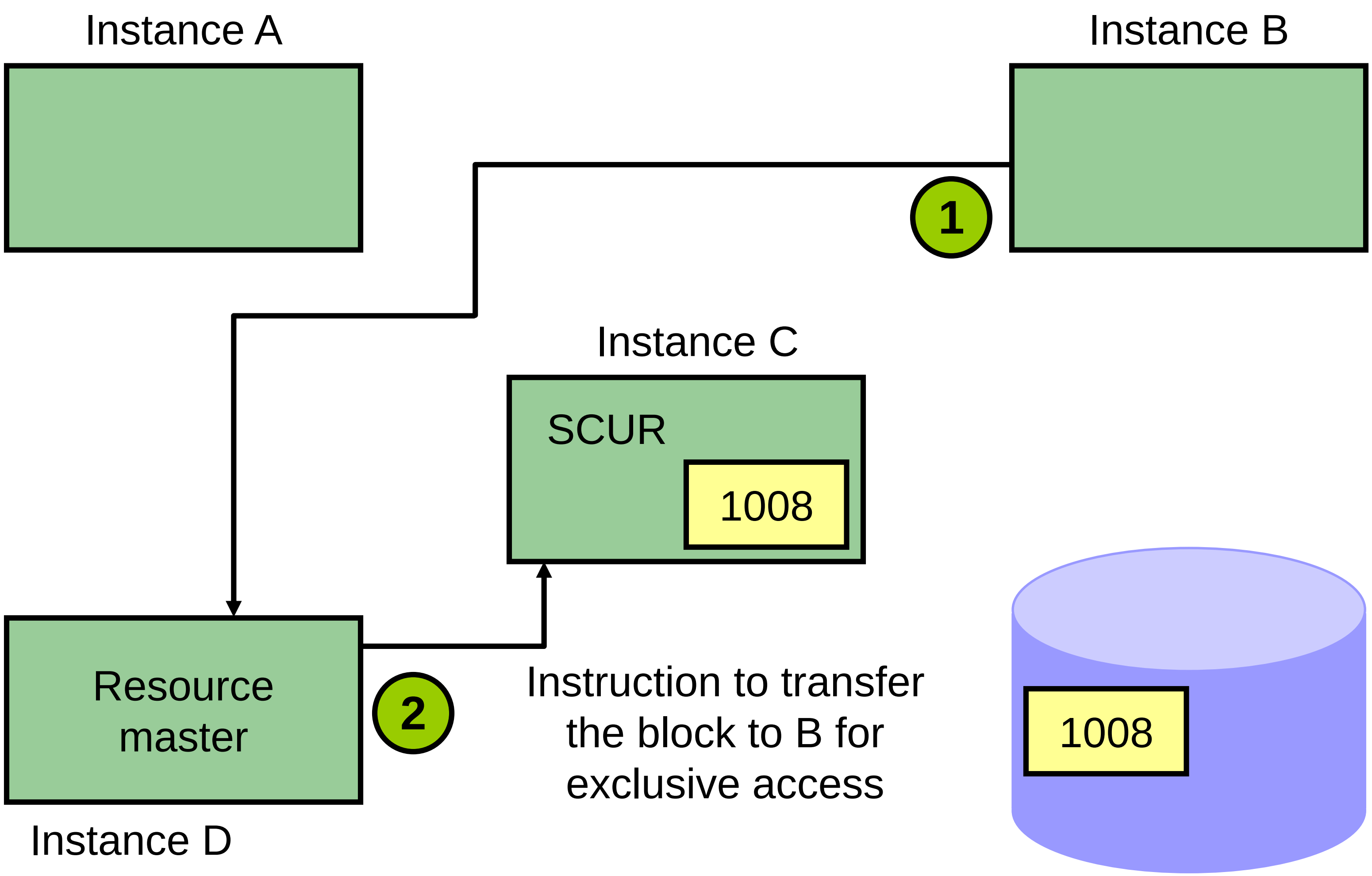
Scenario 1: Read from Disk



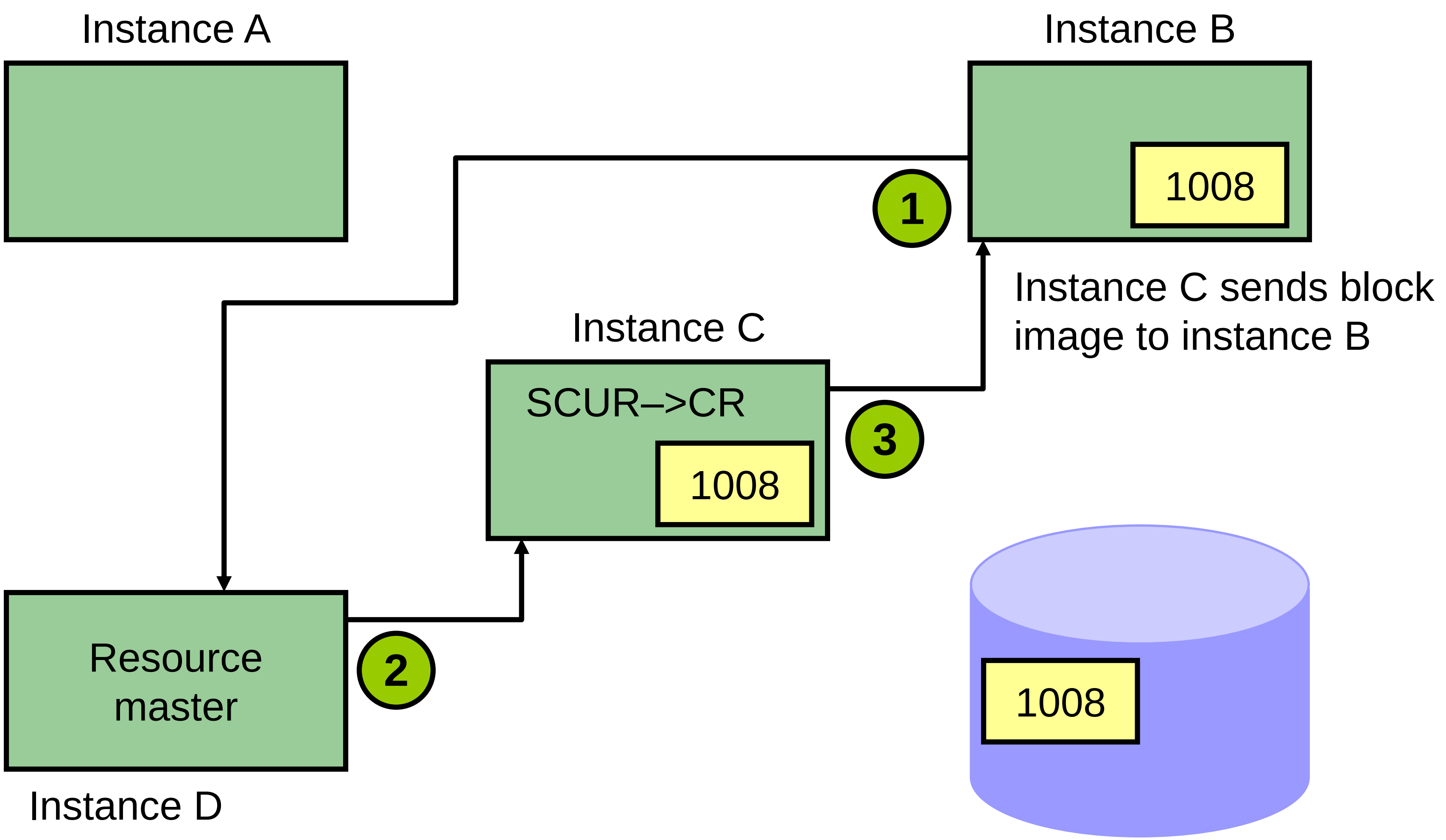
Scenario 2: Read-Write Cache Fusion



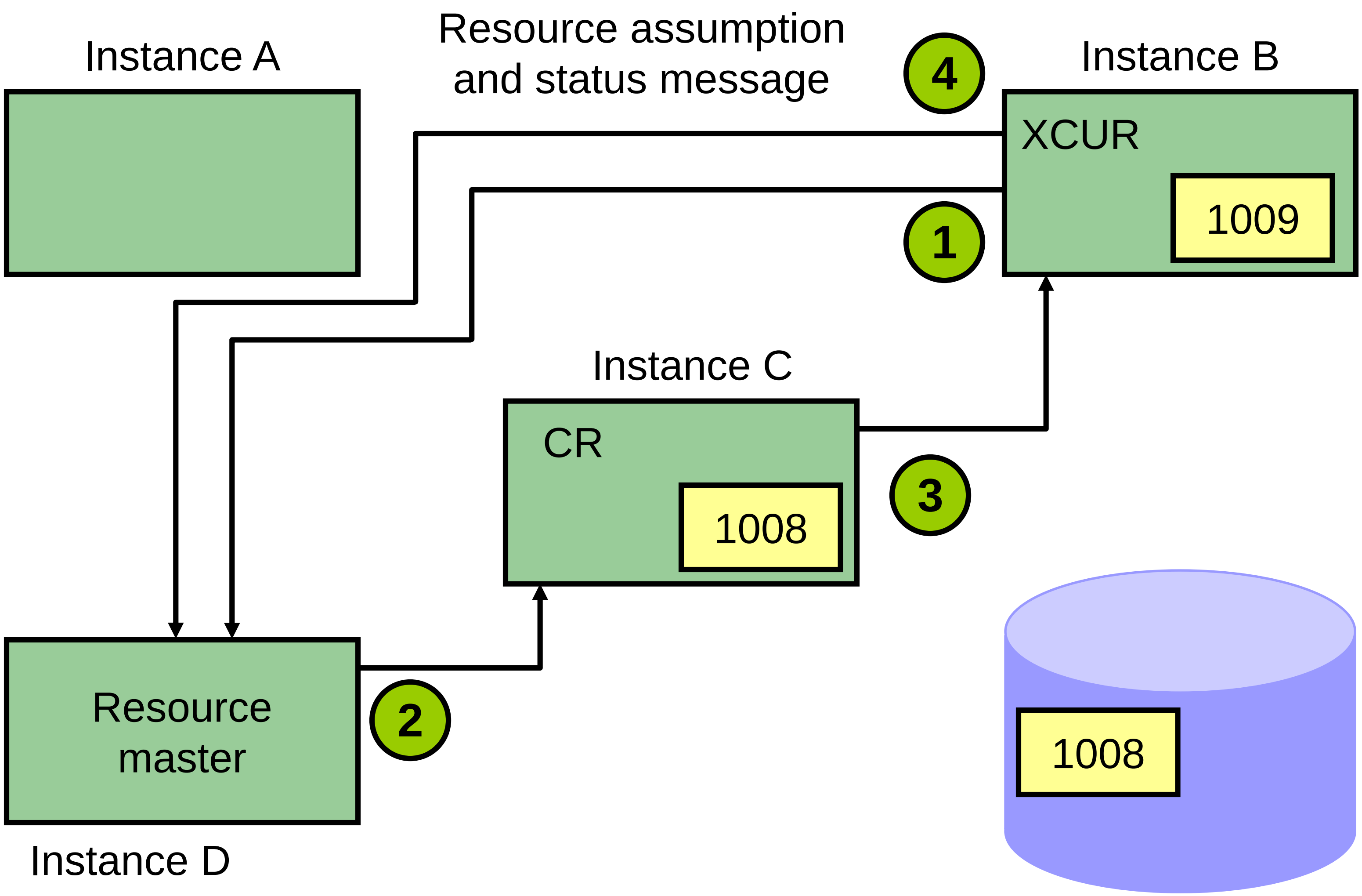
Scenario 2: Read-Write Cache Fusion



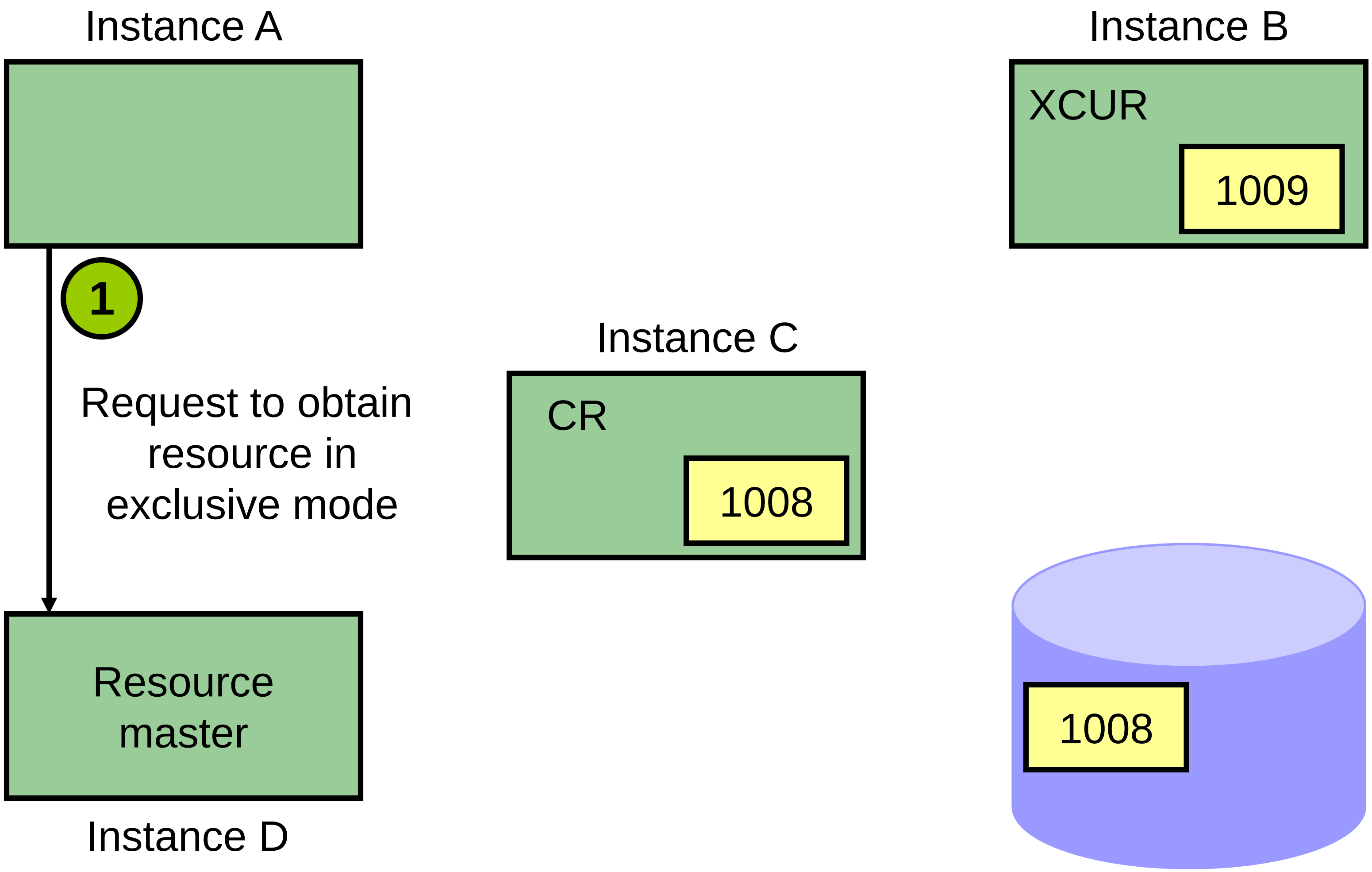
Scenario 2: Read-Write Cache Fusion



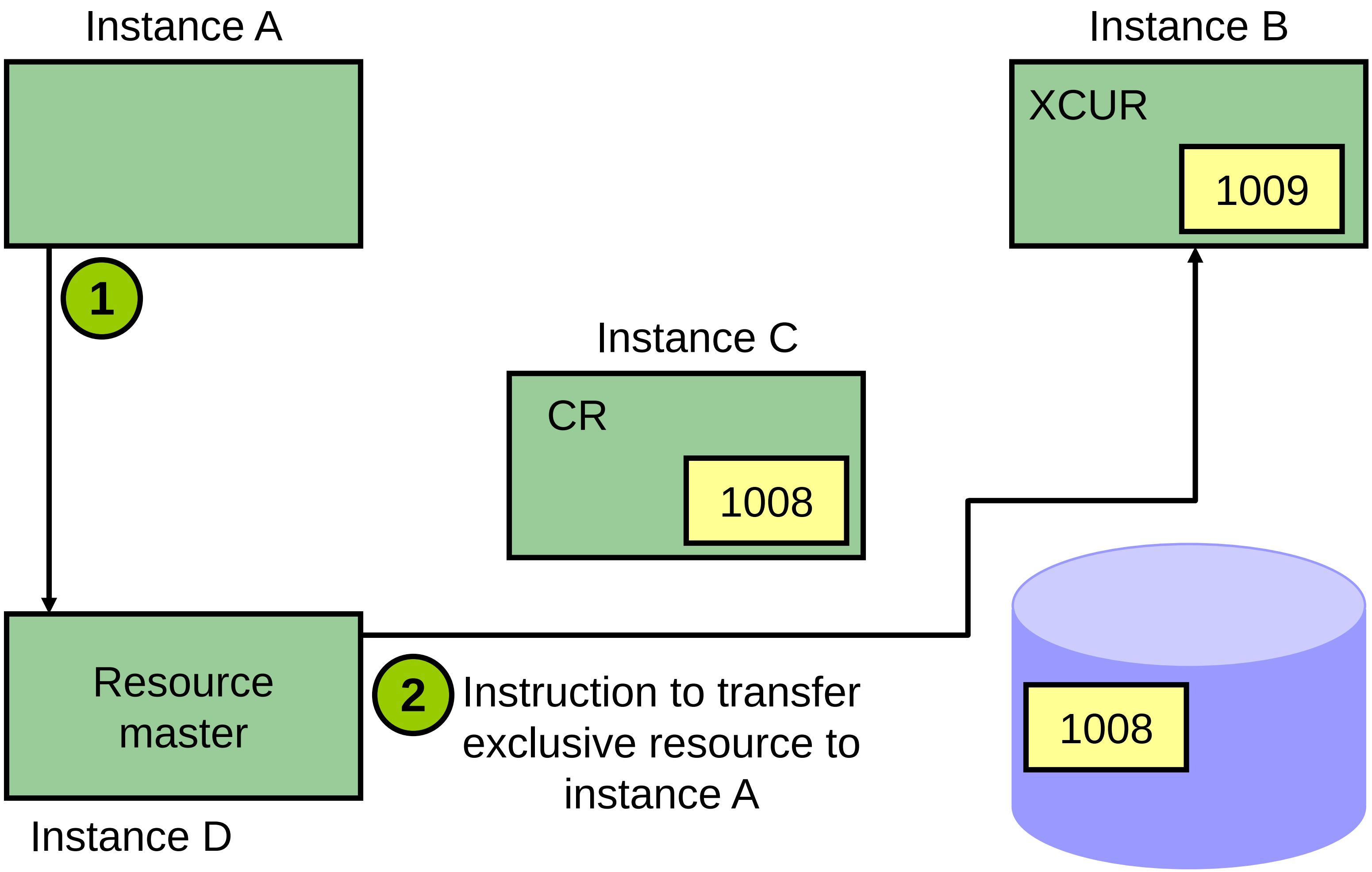
Scenario 2: Read-Write Cache Fusion



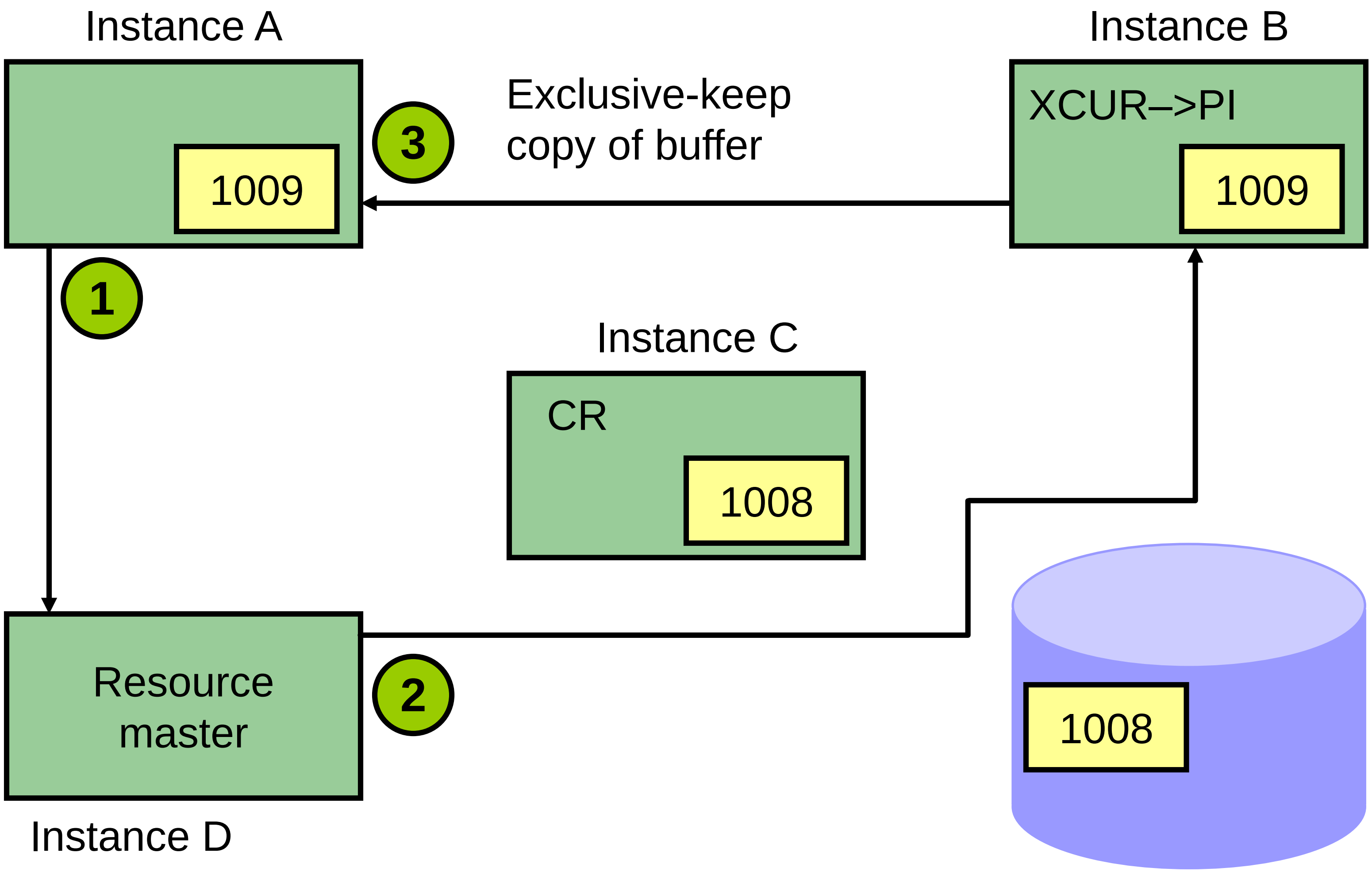
Scenario 3: Write-Write Cache Fusion



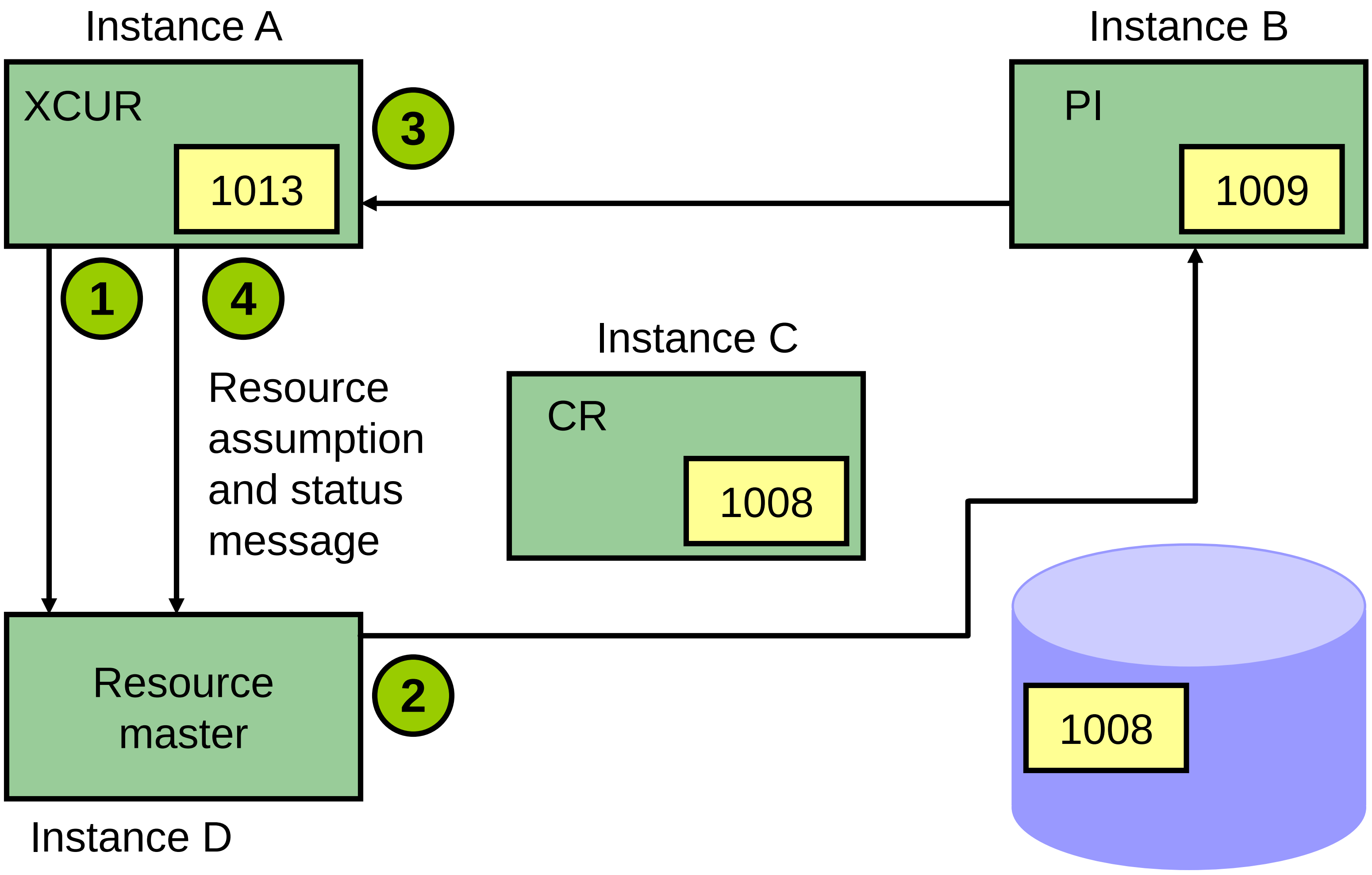
Scenario 3: Write-Write Cache Fusion



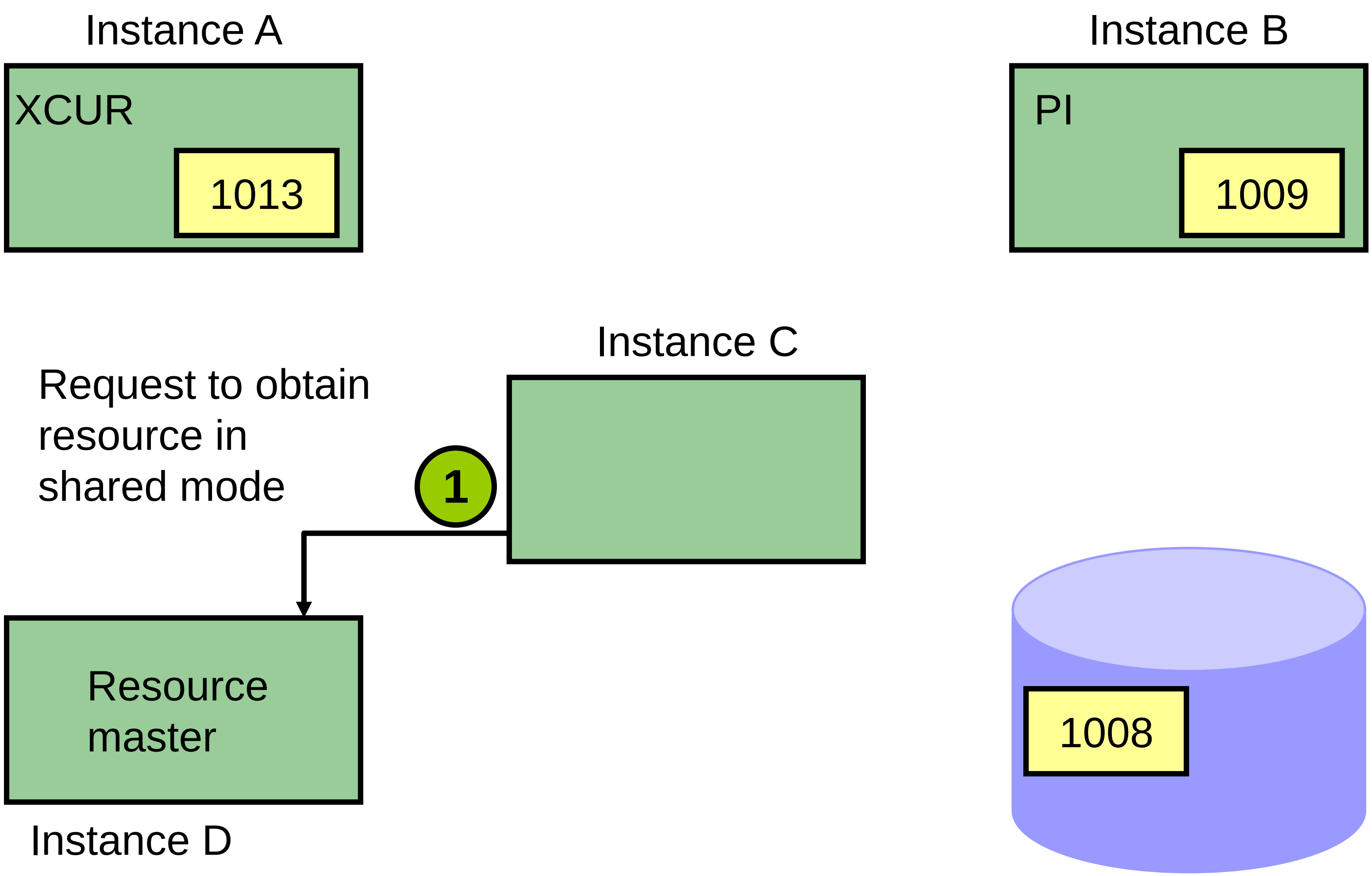
Scenario 3: Write-Write Cache Fusion



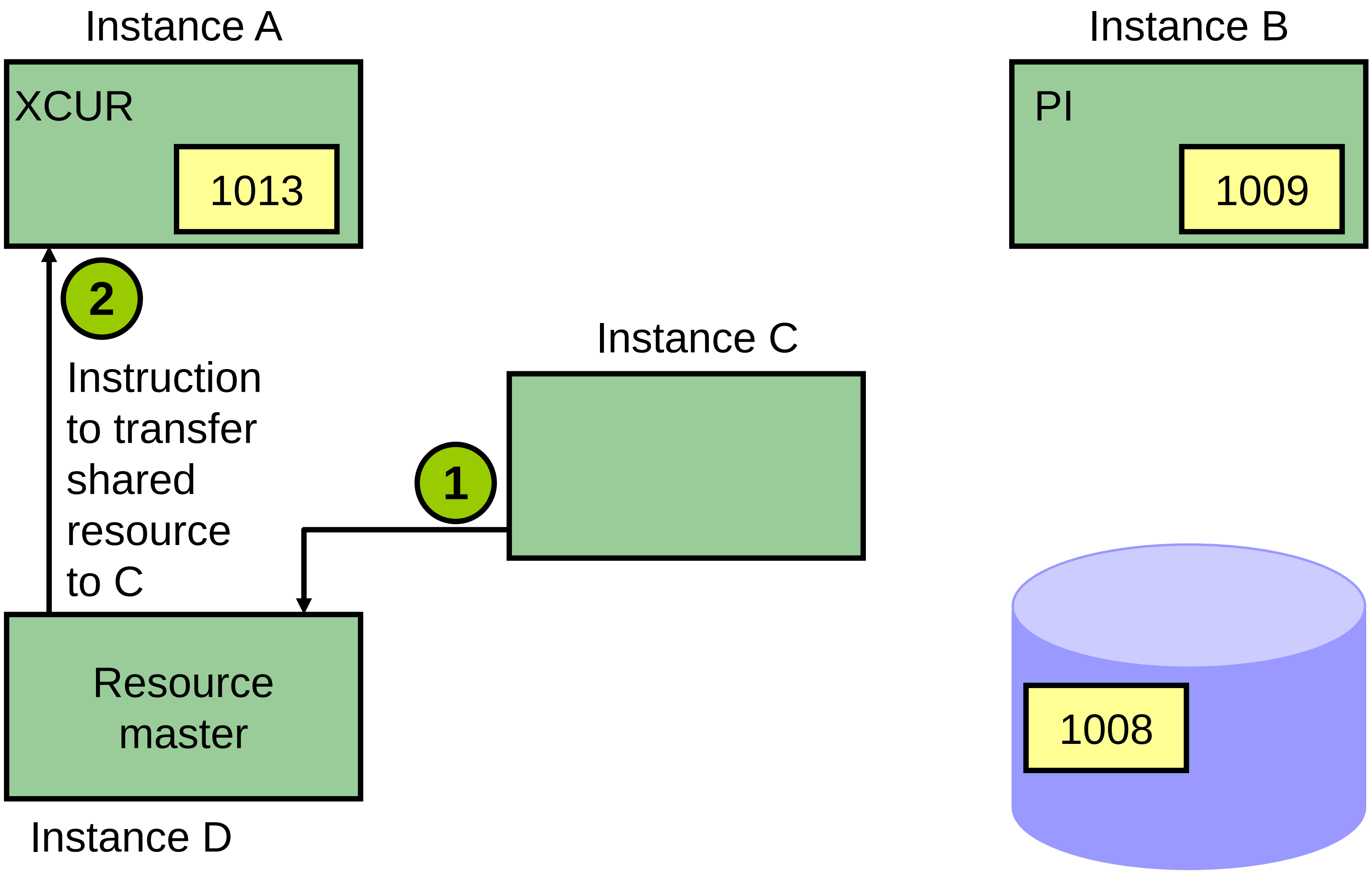
Scenario 3: Write-Write Cache Fusion



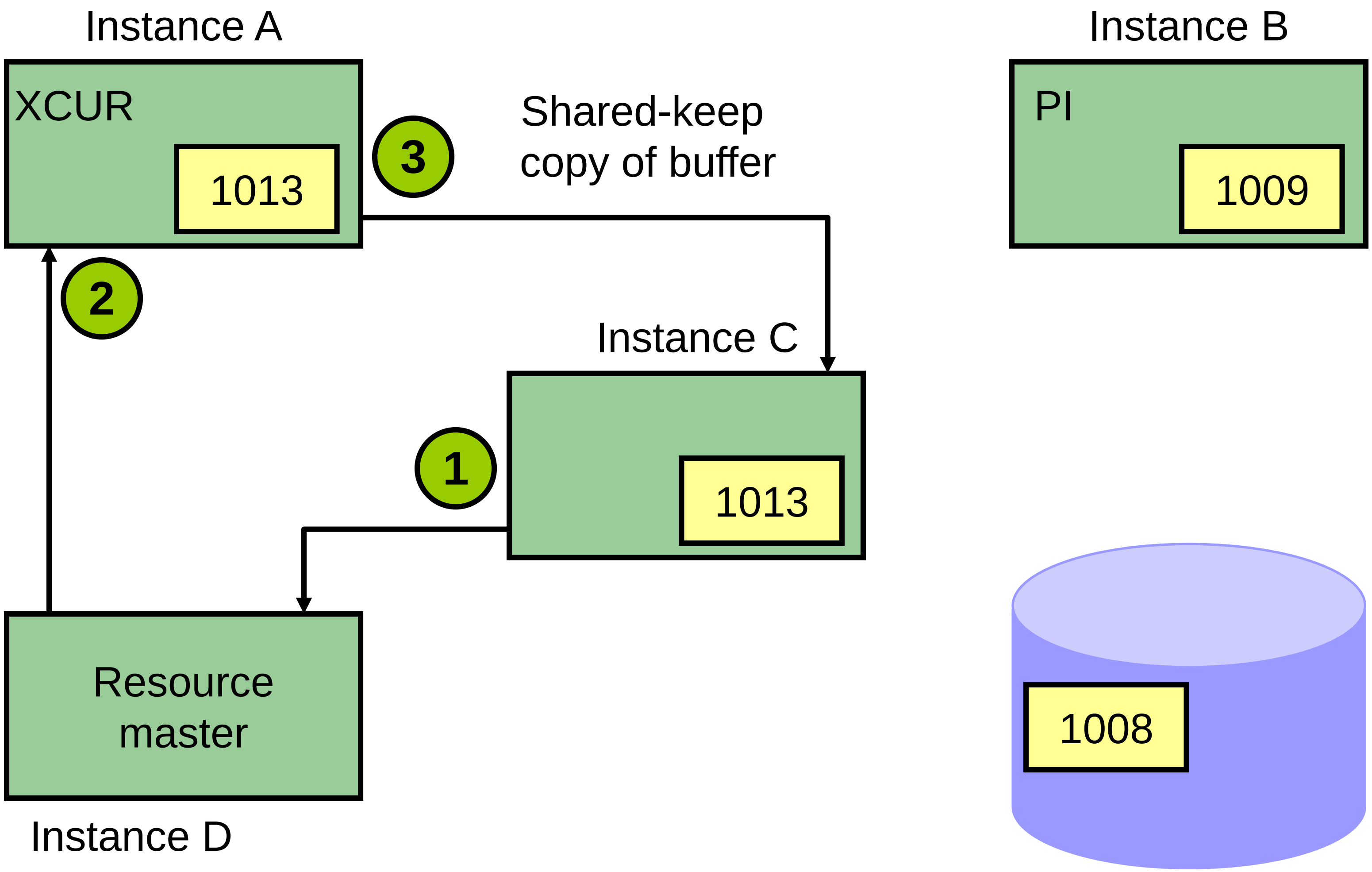
Scenario 4: Write-Read Cache Fusion



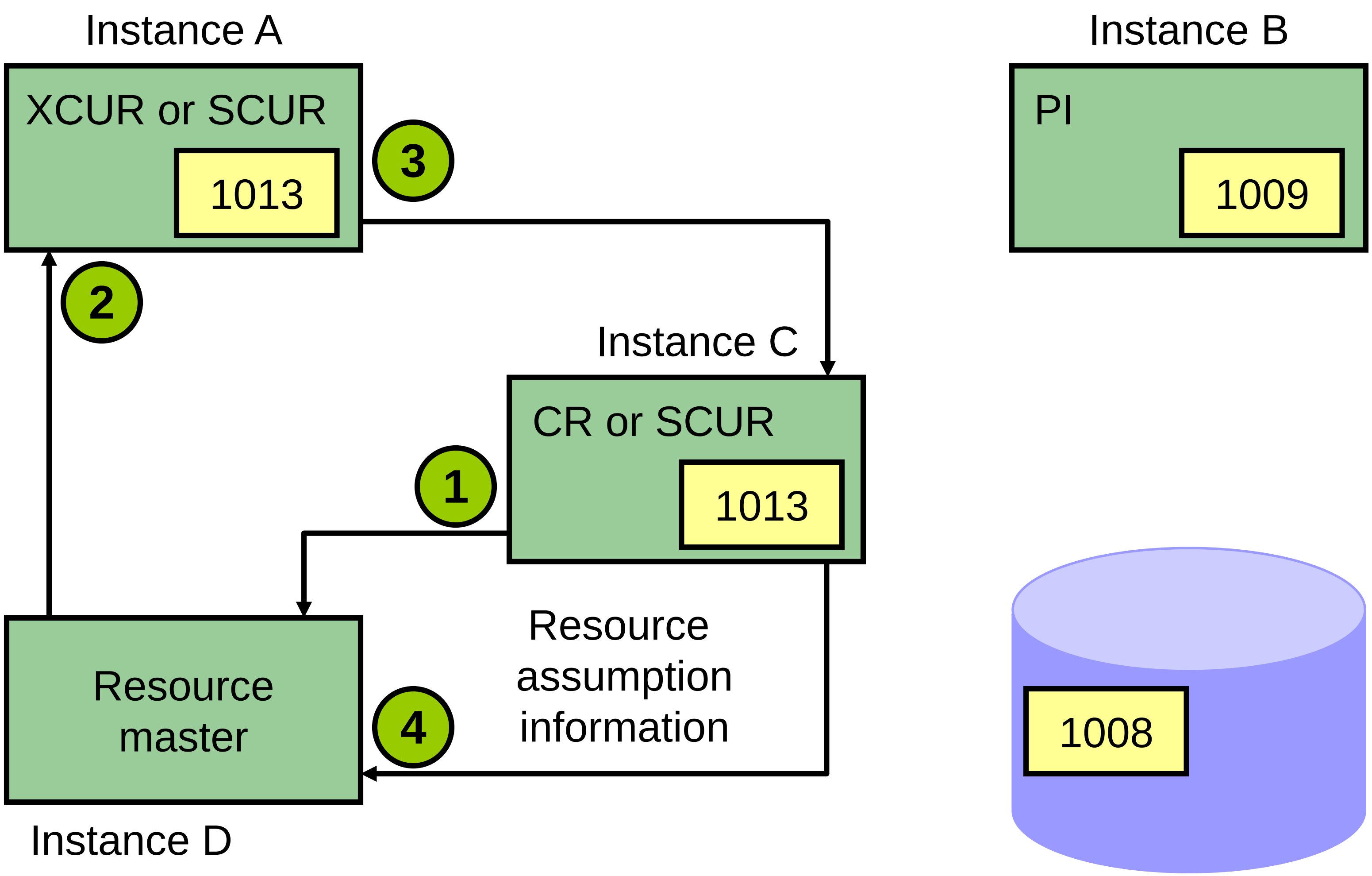
Scenario 4: Write-Read Cache Fusion



Scenario 4: Write-Read Cache Fusion



Scenario 4: Write-Read Cache Fusion



Global Cache Management Scenarios for Multi-Block Reads

- When multi-block read requests occur:
  - The instance doing the I/O must acquire resources for each block in the correct state
  - LMSn coordination from the requesting instance to the LMSn on the mastering instance(s) happens
  - Different blocks in the same multi-block read may have different mastering instances
  - Dynamic remastering, described earlier, may help reduce the performance overhead
- There are several scenarios for multi-block reads:
  - No resource masters exist for any block.
  - Resource masters for some block(s), all are `SCUR`
  - Resource masters for some block(s), some are `XCUR`

Useful Global Resource Management Views

- GV$SESSION_WAIT
- GV$SYSSTAT
- GV$GES_STATISTICS
- V$RESOURCE_LIMIT
- V$BH
- GV$LOCK
- V$CR_BLOCK_SERVER
- V$CURRENT_BLOCK_SERVER
- V$INSTANCE_CACHE_TRANSFER
- V$DYNAMIC_REMASTER_STATS
- GV$RESULT_CACHE_STATS
- V$GCSPFMASTER_INFO

Quiz

Which statement about the Global Resource Directory is *not* true?

a. Resource metadata is held in the Global Resource Directory (`GRD`).
b. An object under global concurrency control is called an asset.
c. Global enqueue resources are used for enqueues and locks.
d. Global cache resources are used for buffer cache control.
e. The `GRD` is distributed among all active instances of each database or ASM environment.

Summary

In this lesson, you should have learned how to describe:

- The need for global concurrency control
- Global Resource Directory
- How global resources are managed
- RAC global resource access coordination
  - Global enqueue and instance lock management
  - Global buffer cache management

Practice 7: Overview

This practice covers the following topics:

- Buffer Cache Fusion
- Row Cache Coordination for Sequences
- Library Cache Coordination for Procedure and Packages
- Global Enqueue Coordination
- Results Cache Coordination

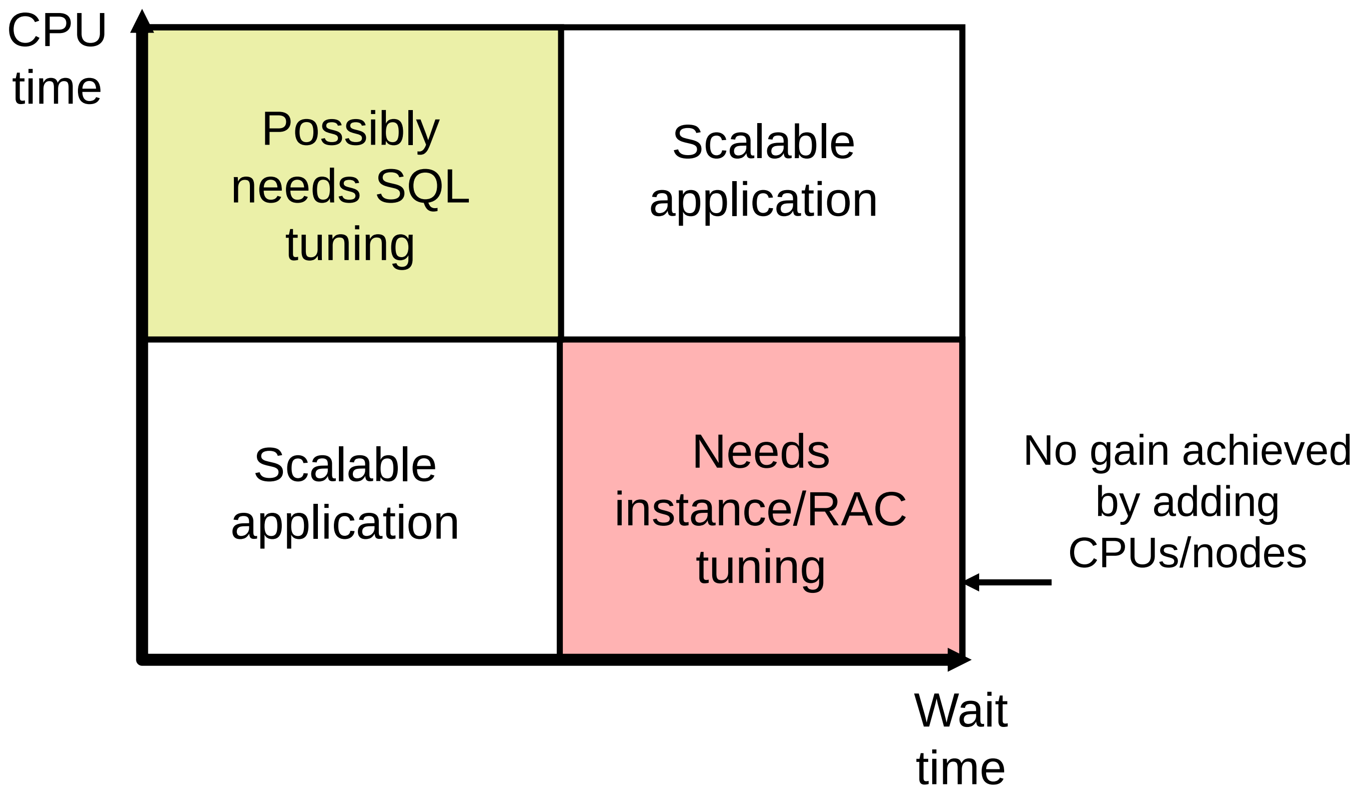
RAC Database Monitoring and Tuning

Objectives

After completing this lesson, you should be able to:

- Determine RAC-specific tuning components
- Determine RAC-specific wait events, global enqueues, and system statistics
- Implement the most common RAC tuning tips
- Use the Cluster Database Performance pages
- Use the Automatic Workload Repository (AWR) in RAC
- Use Automatic Database Diagnostic Monitor (ADDM) in RAC

CPU and Wait Time Tuning Dimensions



RAC-Specific Tuning

- Tune for a single instance first.
- Tune for RAC:
  - Instance recovery
  - Interconnect traffic
  - Point of serialization can be exacerbated.
- RAC-reactive tuning tools:
  - Specific wait events
  - System and enqueue statistics
  - Enterprise Manager performance pages
  - Statspack and AWR reports

  Certain combinations are characteristic of well-known tuning cases.
- RAC-proactive tuning tools:
  - AWR snapshots
  - ADDM reports

Analyzing Cache Fusion Impact in RAC

- The cost of block access and cache coherency is represented by:
  - Global Cache Services statistics
  - Global Cache Services wait events
- The response time for cache fusion transfers is determined by:
  - Overhead of the physical interconnect components
  - IPC protocol
  - GCS protocol
- The response time is not generally affected by disk I/O factors.

Typical Latencies for RAC Operations

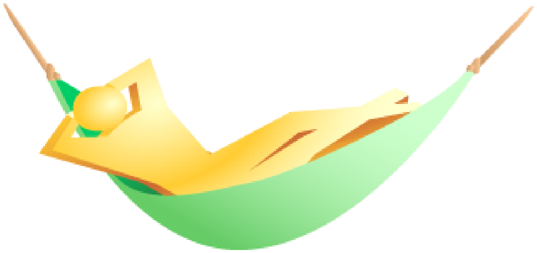
AWR Report Latency Name	Lower Bound	Typical	Upper Bound
Average time to process cr block request	0.1	1	10
Avg global cache cr block receive time (ms)	0.3	4	12
Average time to process current block request	0.1	3	23
Avg global cache current block receive time(ms)	0.3	8	30

Global Cache and Enqueue Workload Characteristics

		CR Blocks					CU Blocks				
I#	GE Get Time (ms)	Receive Time (ms)	Build Time (ms)	Send Time (ms)	Flush Time (ms)	Log Flush CR Srvd %	Receive Time (ms)	Pin Time (ms)	Send Time (ms)	Flush Time (ms)	Log Flush CU Srvd %
3	2.03	7.46	0.00	0.00	30.72	2.68	6.38	0.11	0.00	22.30	5.72

Wait Events for RAC

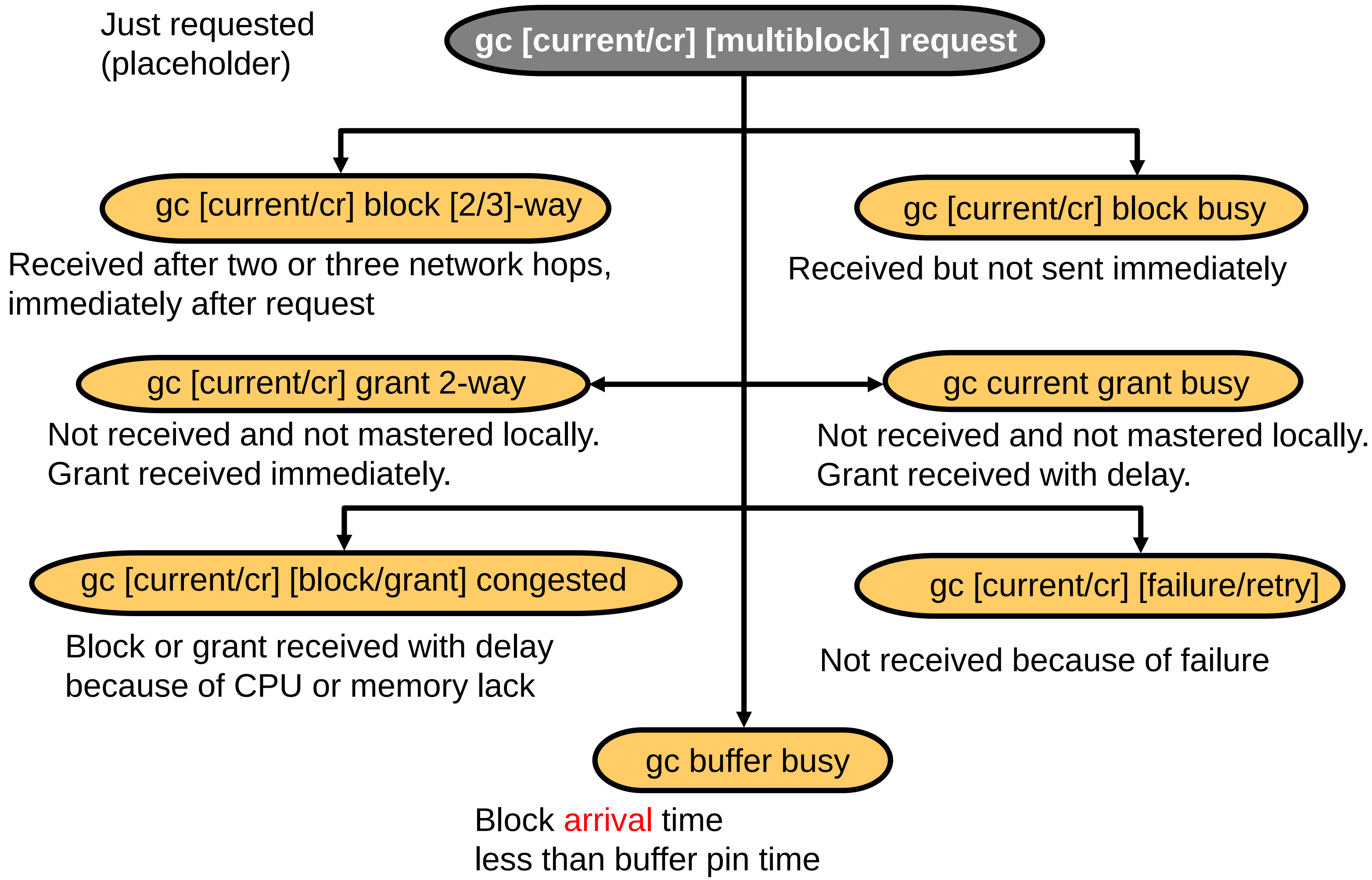
- Wait events help to analyze what sessions are waiting for.
- Wait times are attributed to events that reflect the outcome of a request:
  - Placeholders while waiting
  - Precise events after waiting
- Global cache waits are summarized in a broader category called Cluster Wait Class.
- These wait events are used in ADDM to enable cache fusion diagnostics.



Wait Event Views

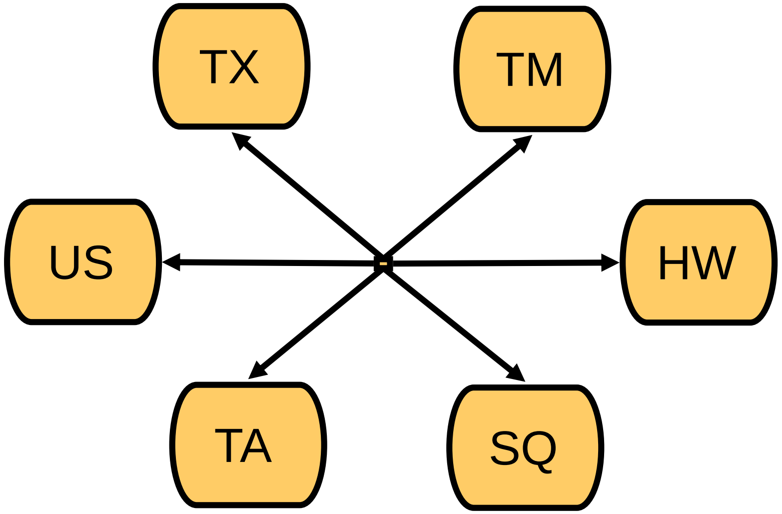
Total waits for an event	V$SYSTEM_EVENT
Waits for a wait event class by a session	V$SESSION_WAIT_CLASS
Waits for an event by a session	V$SESSION_EVENT
Activity of recent active sessions	V$ACTIVE_SESSION_HISTORY
Last 10 wait events for each active session	V$SESSION_WAIT_HISTORY
Events for which active sessions are waiting	V$SESSION_WAIT
Identify SQL statements impacted by interconnect latencies	V$SQLSTATS

Global Cache Wait Events: Overview



Global Enqueue Waits

- Enqueues are synchronous.
- Enqueues are global resources in RAC.
- The most frequent waits are for:



- The waits may constitute serious serialization points.

Session and System Statistics

- Use V$SYSSTAT to characterize the workload.
- Use V$SESSTAT to monitor important sessions.
- V$SEGMENT_STATISTICS includes RAC statistics.
- RAC-relevant statistic groups are:
  - Global Cache Service statistics
  - Global Enqueue Service statistics
  - Statistics for messages sent
- V$ENQUEUE_STATISTICS determines the enqueue with the highest impact.
- V$INSTANCE_CACHE_TRANSFER breaks down GCS statistics into block classes.

Most Common RAC Tuning Tips

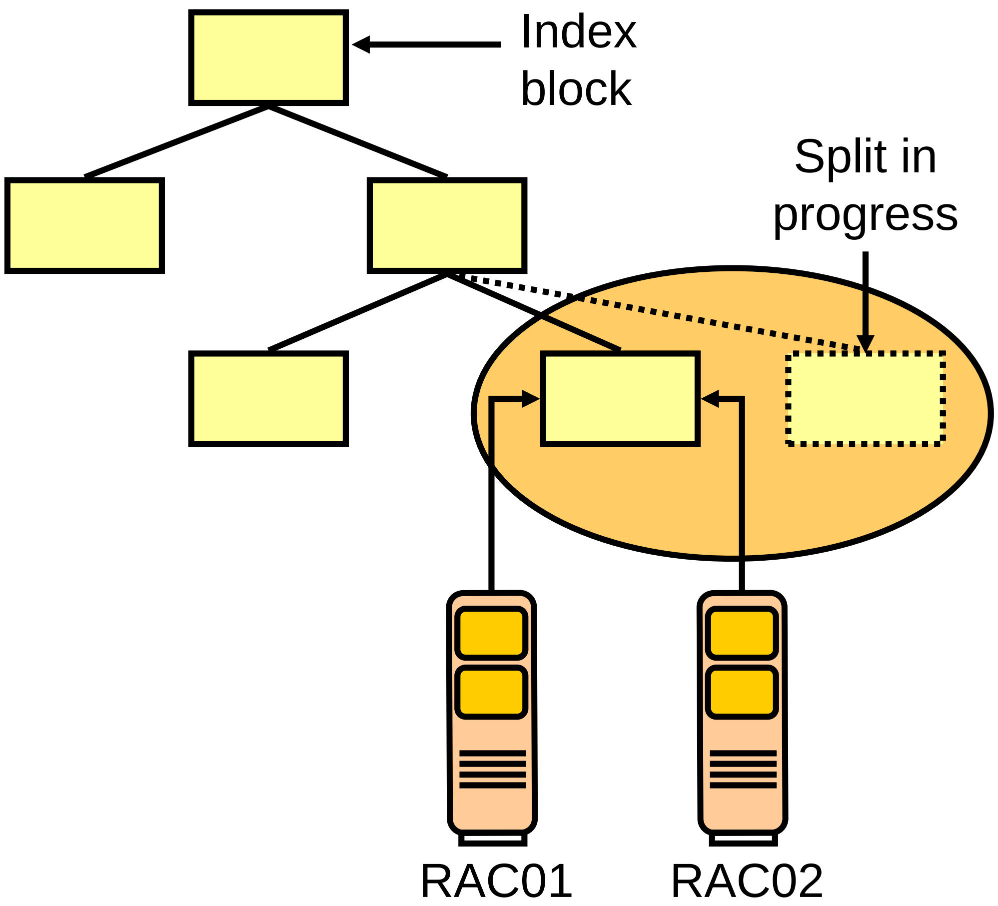
Application tuning is often the most beneficial!

- Reduce long full-table scans in OLTP systems.
- Use Automatic Segment Space Management (ASSM).
- Increase sequence caches.
- Use partitioning to reduce interinstance traffic.
- Avoid unnecessary parsing.
- Minimize locking usage.
- Remove unselective indexes.
- Configure interconnect properly.
- Employ In Memory-Parallel Query

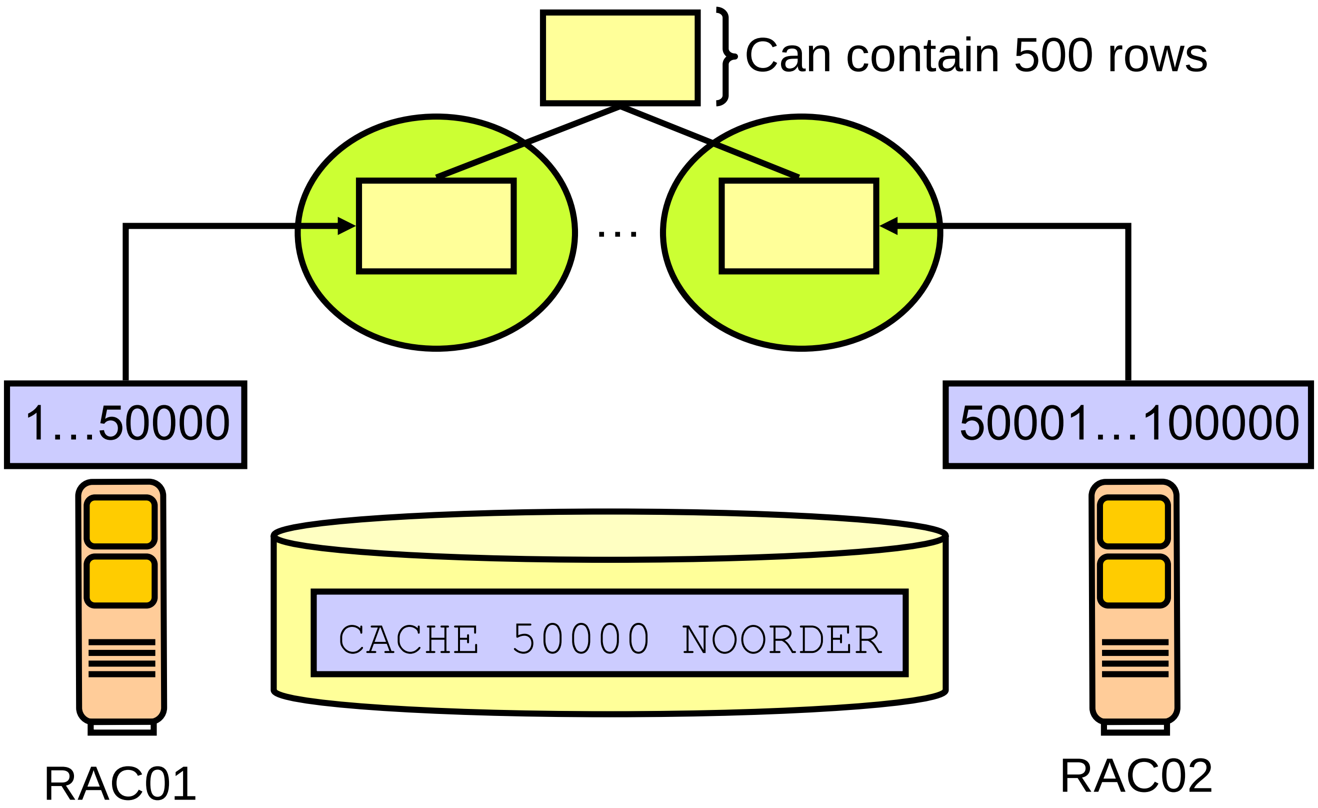
Index Block Contention: Considerations

Wait events
`enq: TX - index contention`
`gc buffer busy`
`gc current block busy`
`gc current split`

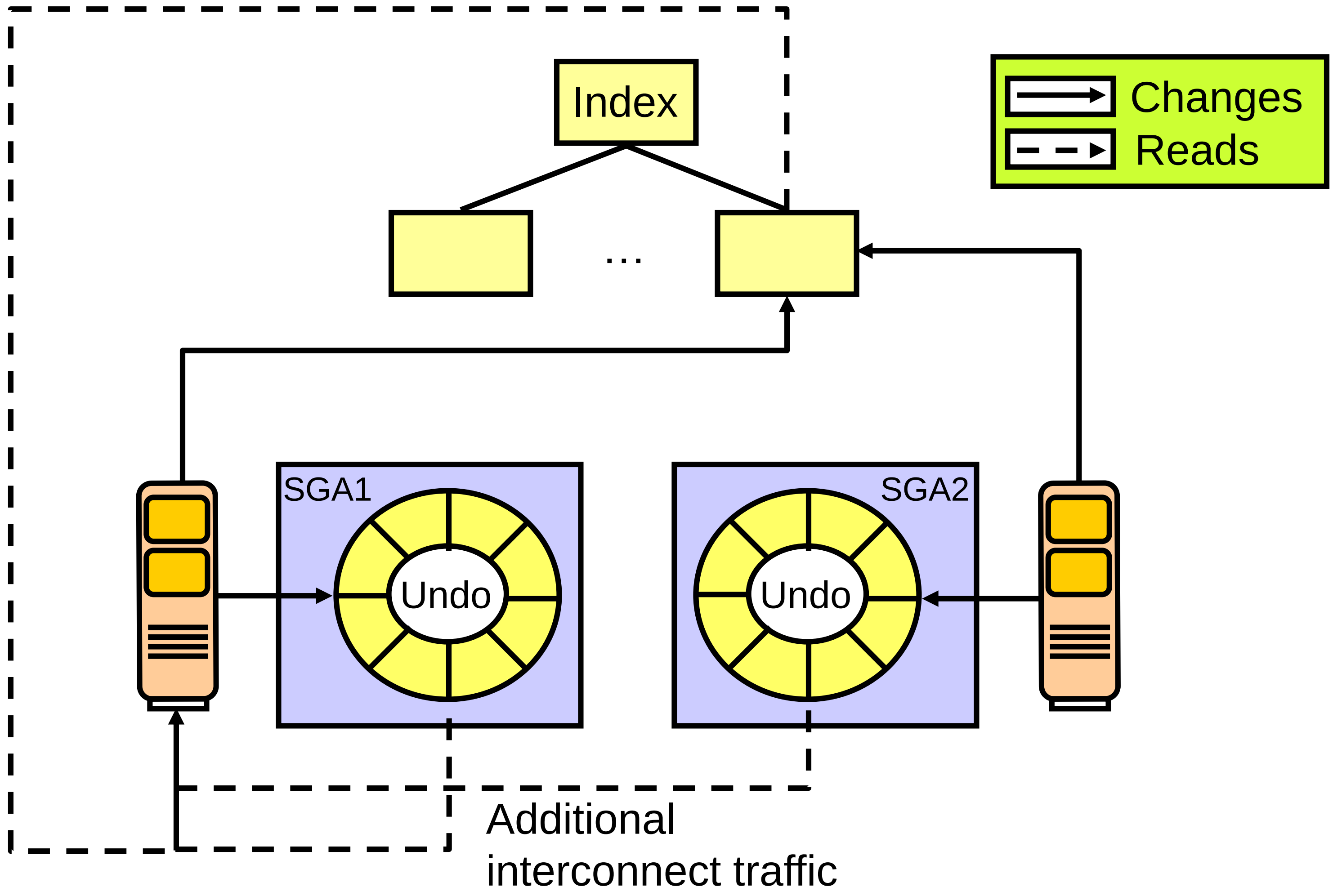
System statistics
`Leaf node splits`
`Branch node splits`
`Exchange deadlocks`
`gcs refuse xid` `gcs ast xid`
`Service ITL waits`



Oracle Sequences and Index Contention

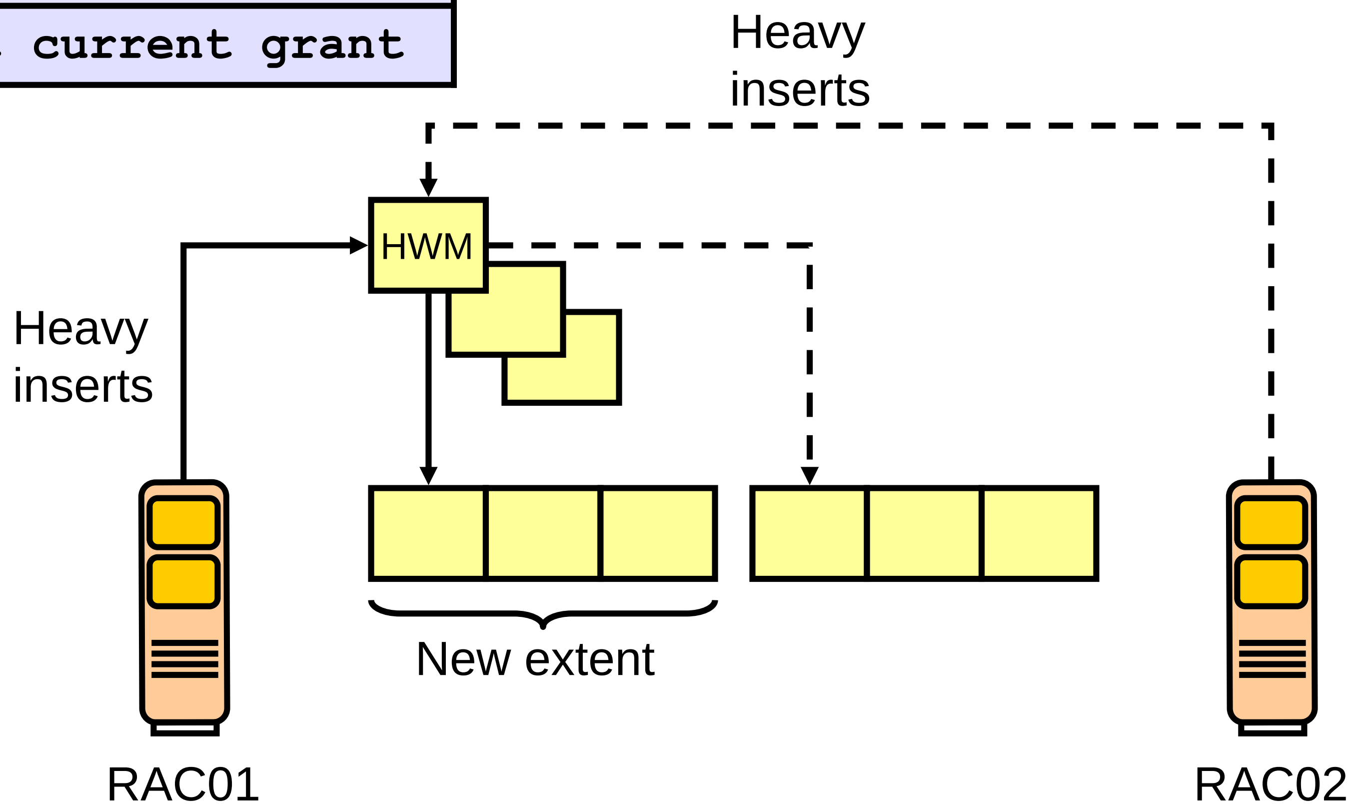


Undo Block Considerations

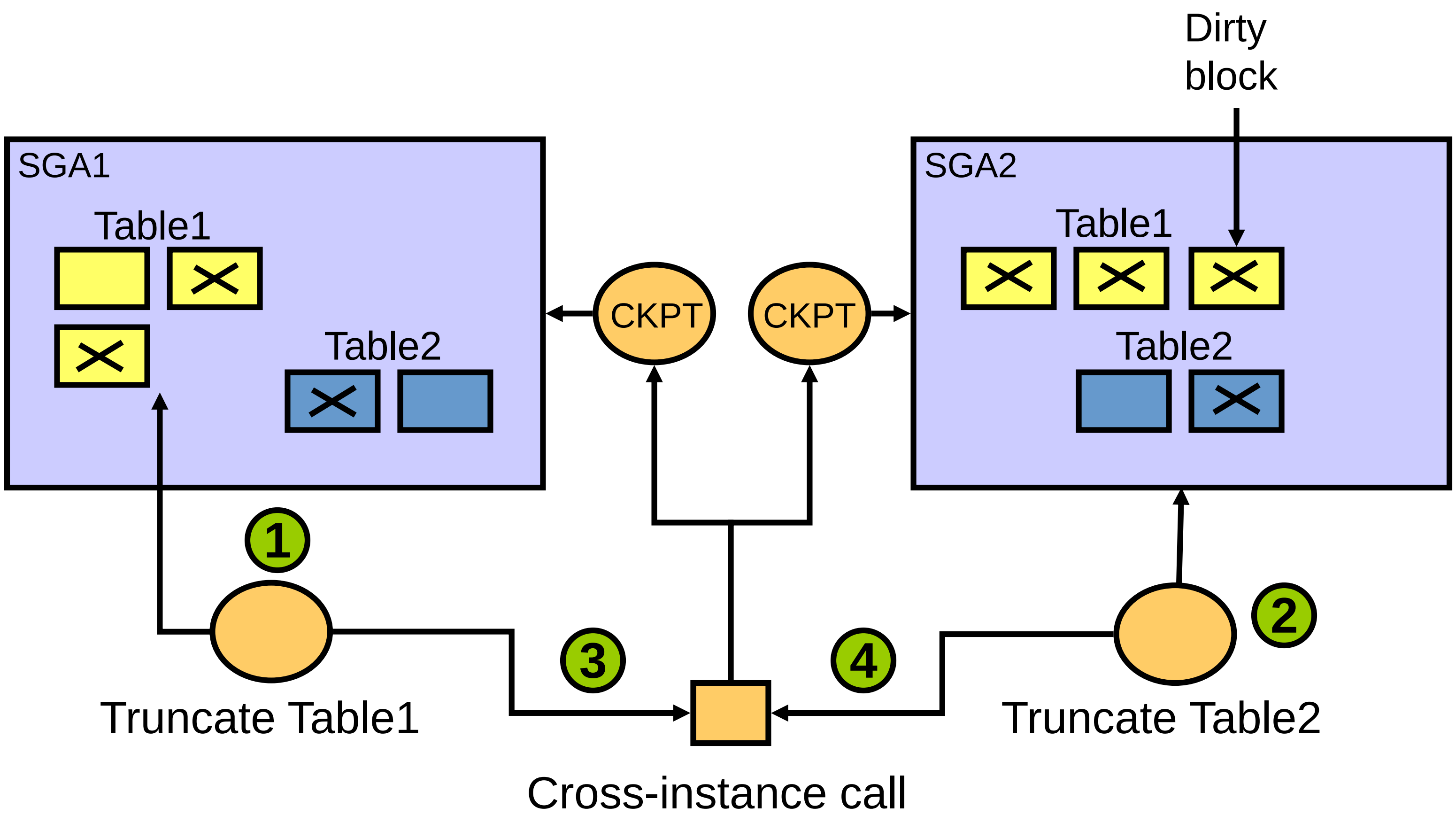


High-Water Mark Considerations

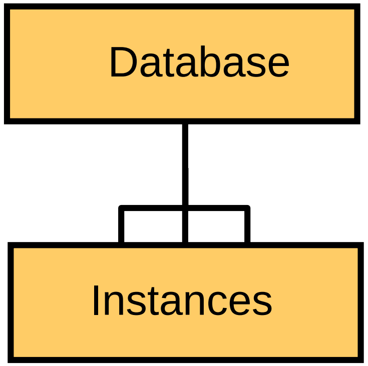
Wait events
`enq: HW - contention`
`gc current grant`



Concurrent Cross-Instance Calls: Considerations



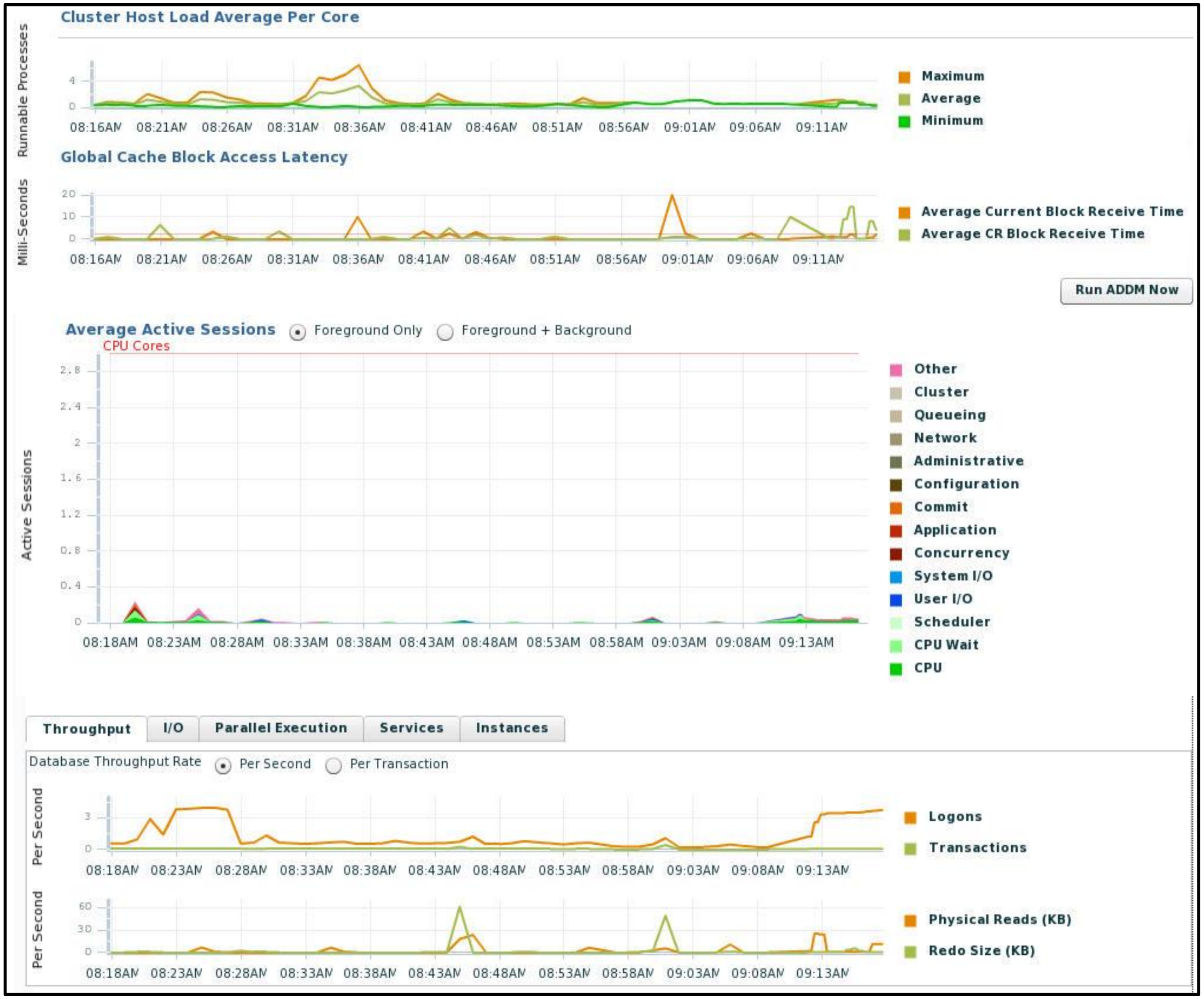
Monitoring RAC Database and Cluster Performance



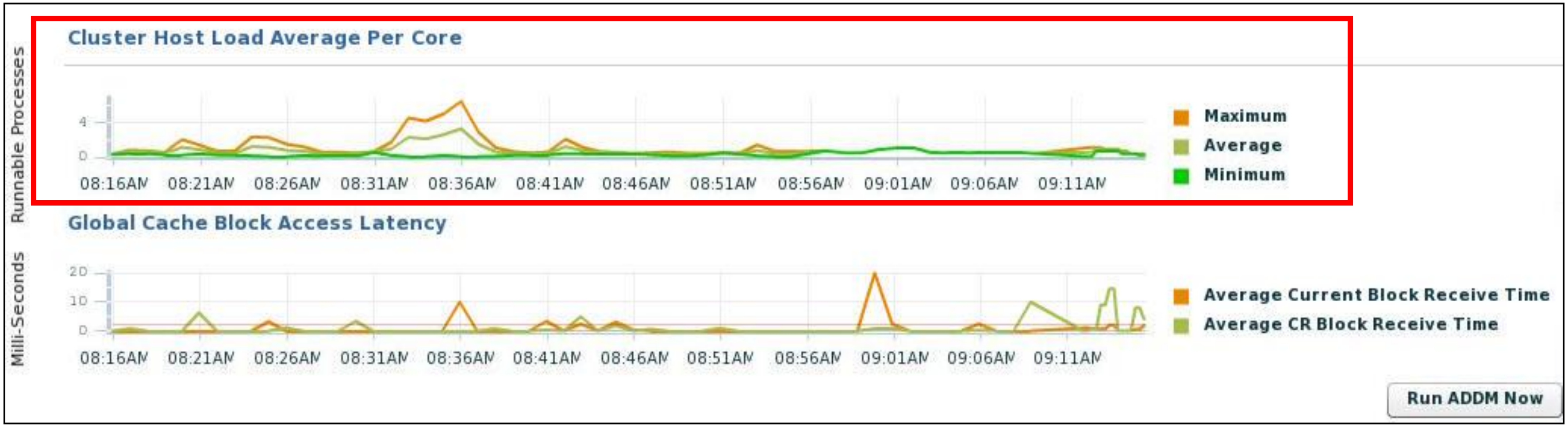
Directly from EM Cloud Control:

- View the status of each node in the cluster.
- View the aggregated alert messages across all the instances.
- Review the issues that are affecting the entire cluster or each instance.
- Monitor the cluster cache coherency statistics.
- Determine whether any of the services for the cluster database are having availability problems.
- Review any outstanding Clusterware interconnect alerts.

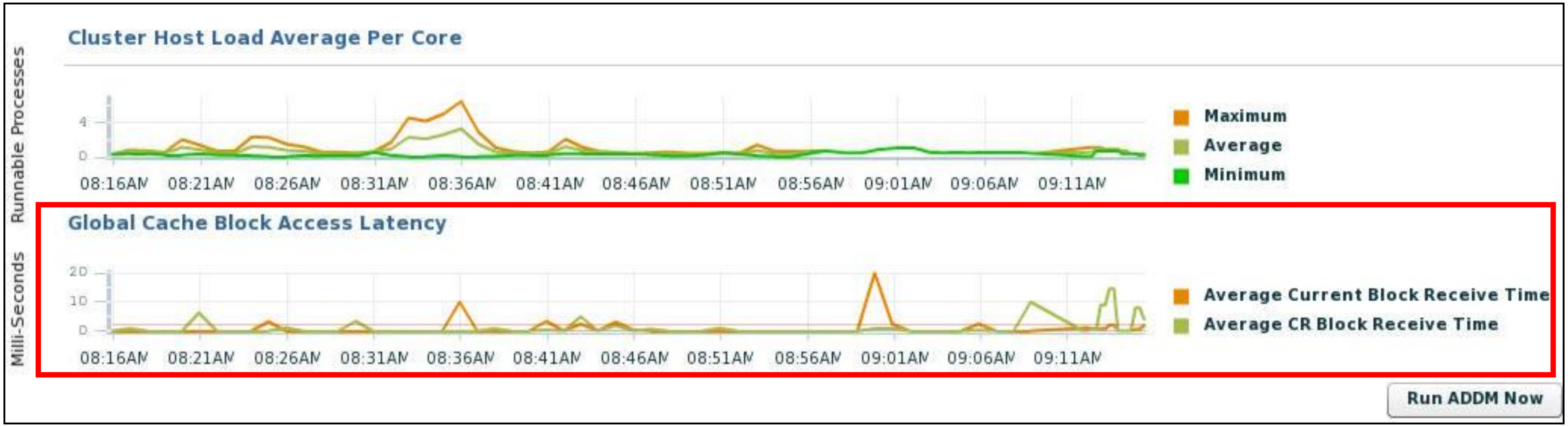
Cluster Database Performance Page



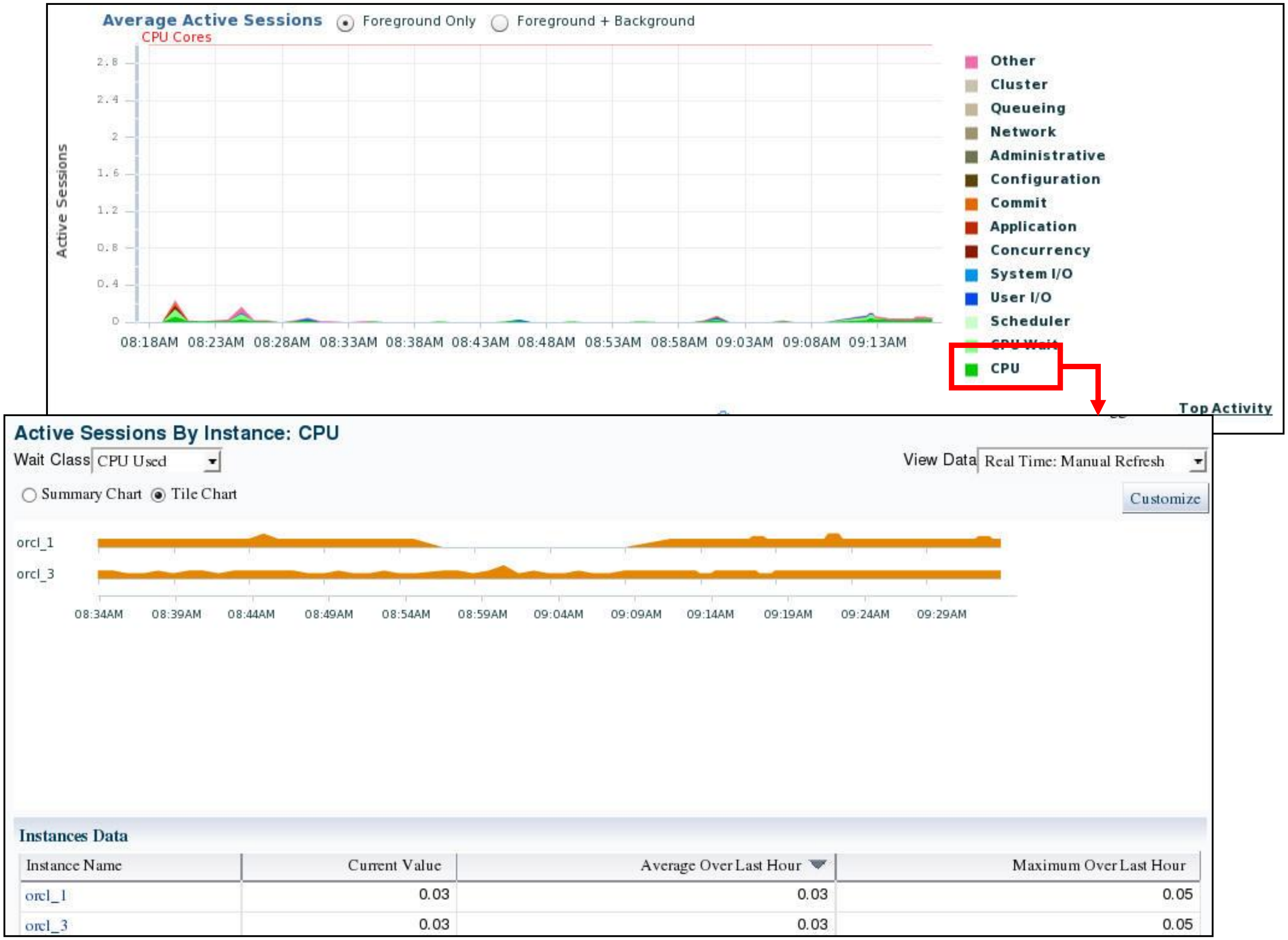
Determining Cluster Host Load Average



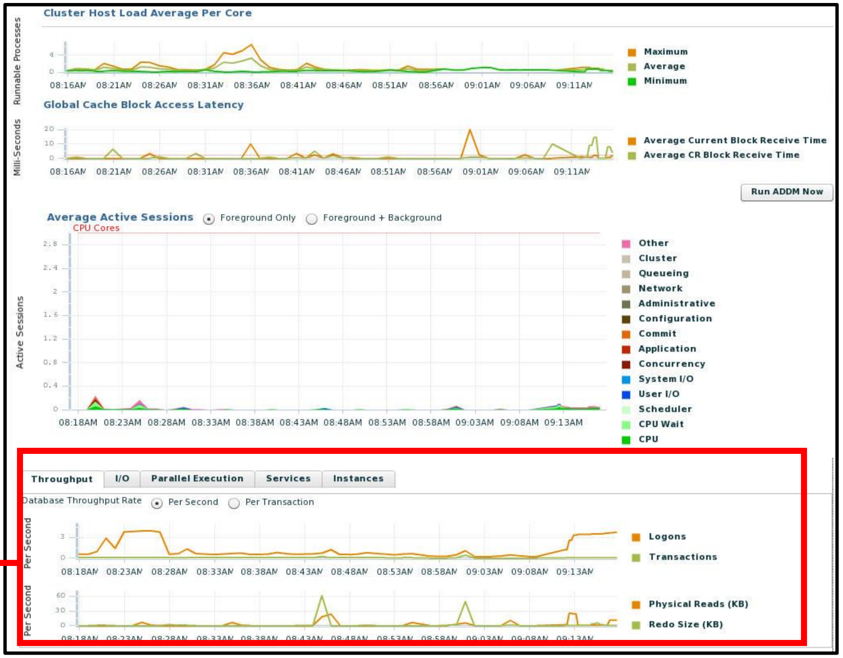
Determining Global Cache Block Access Latency

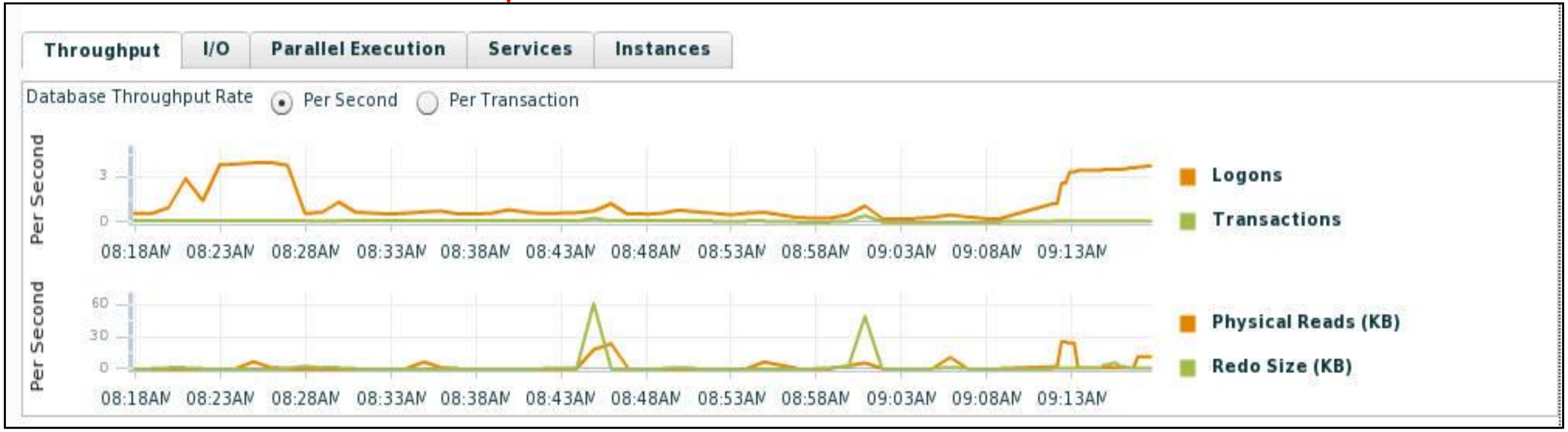


Determining Average Active Sessions

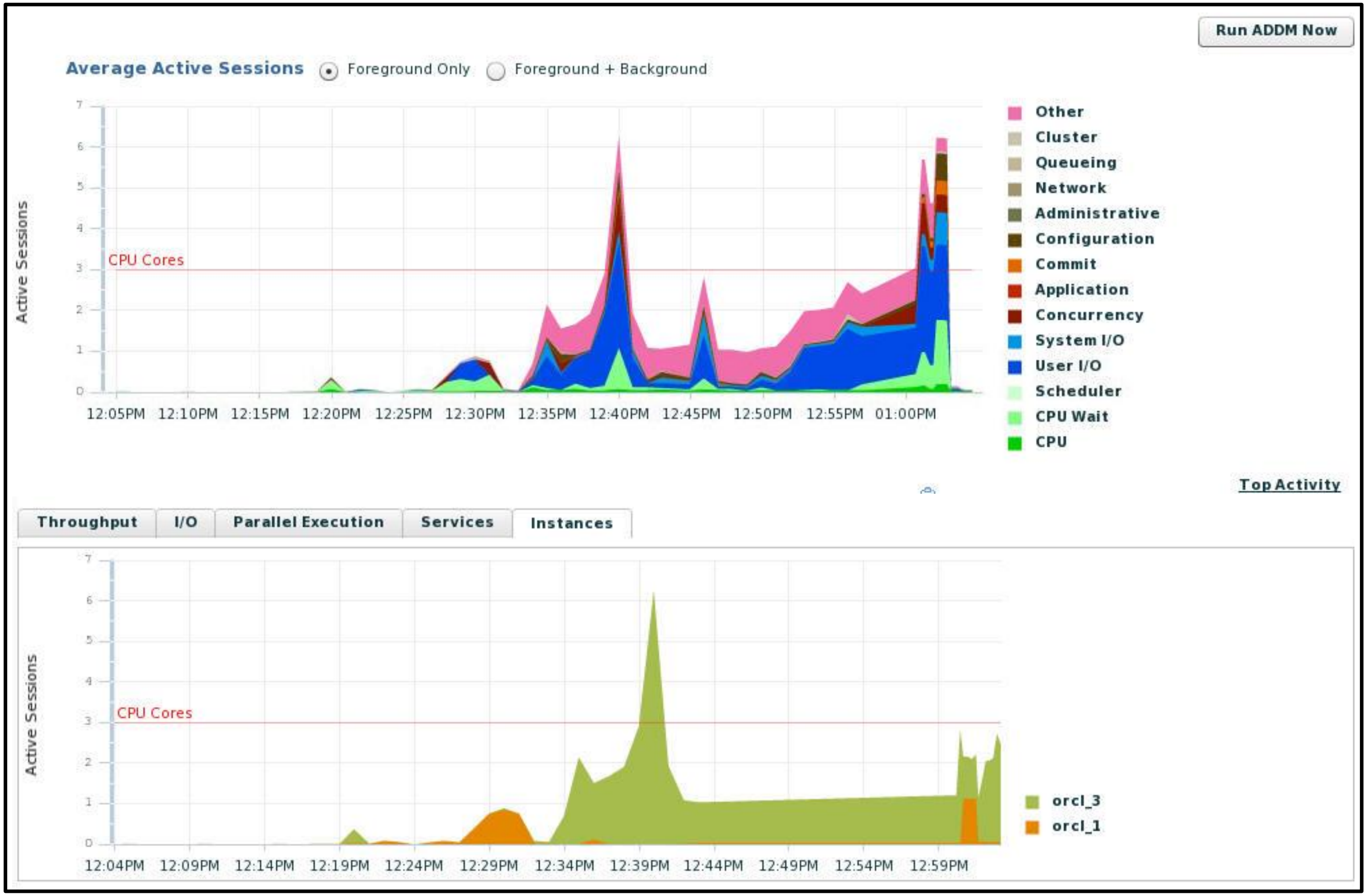


Determining Database Throughput

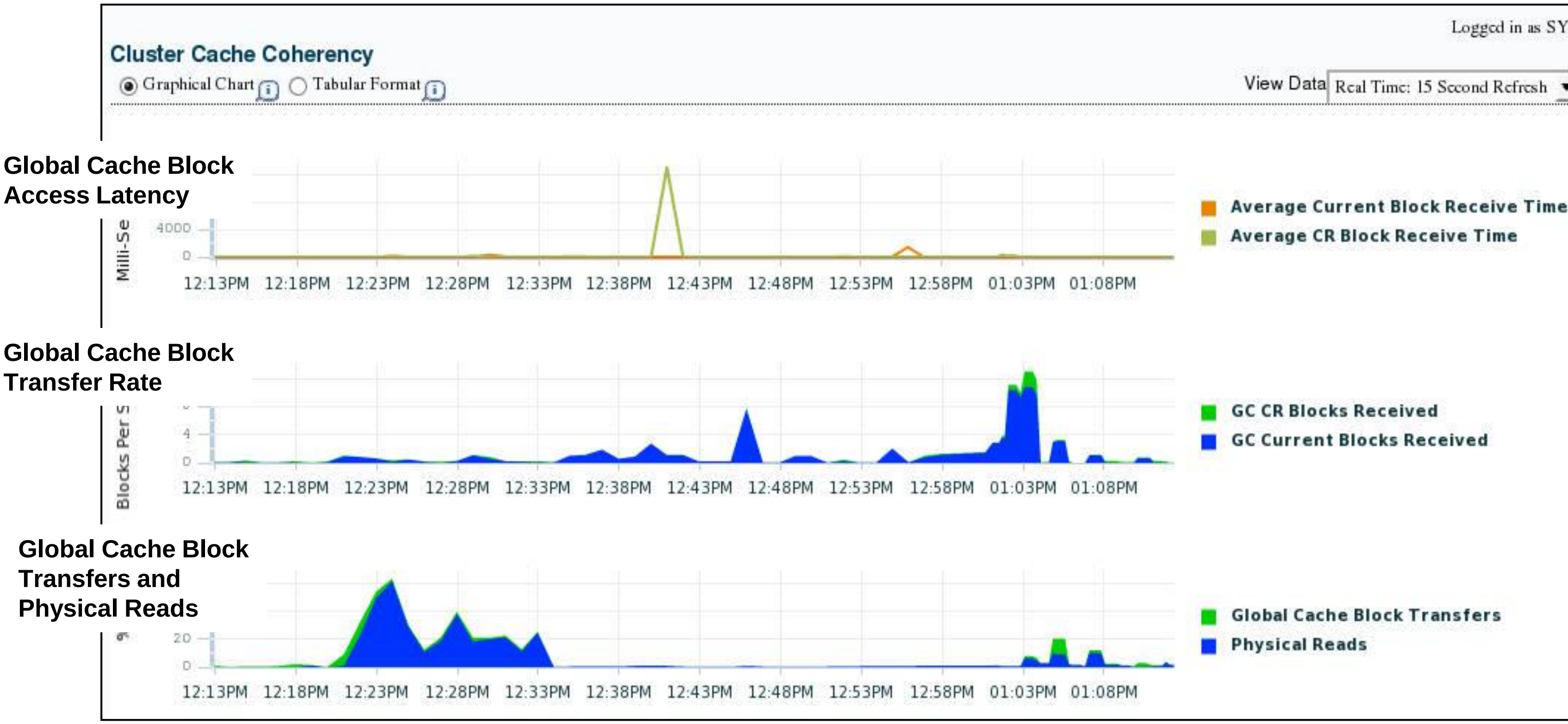




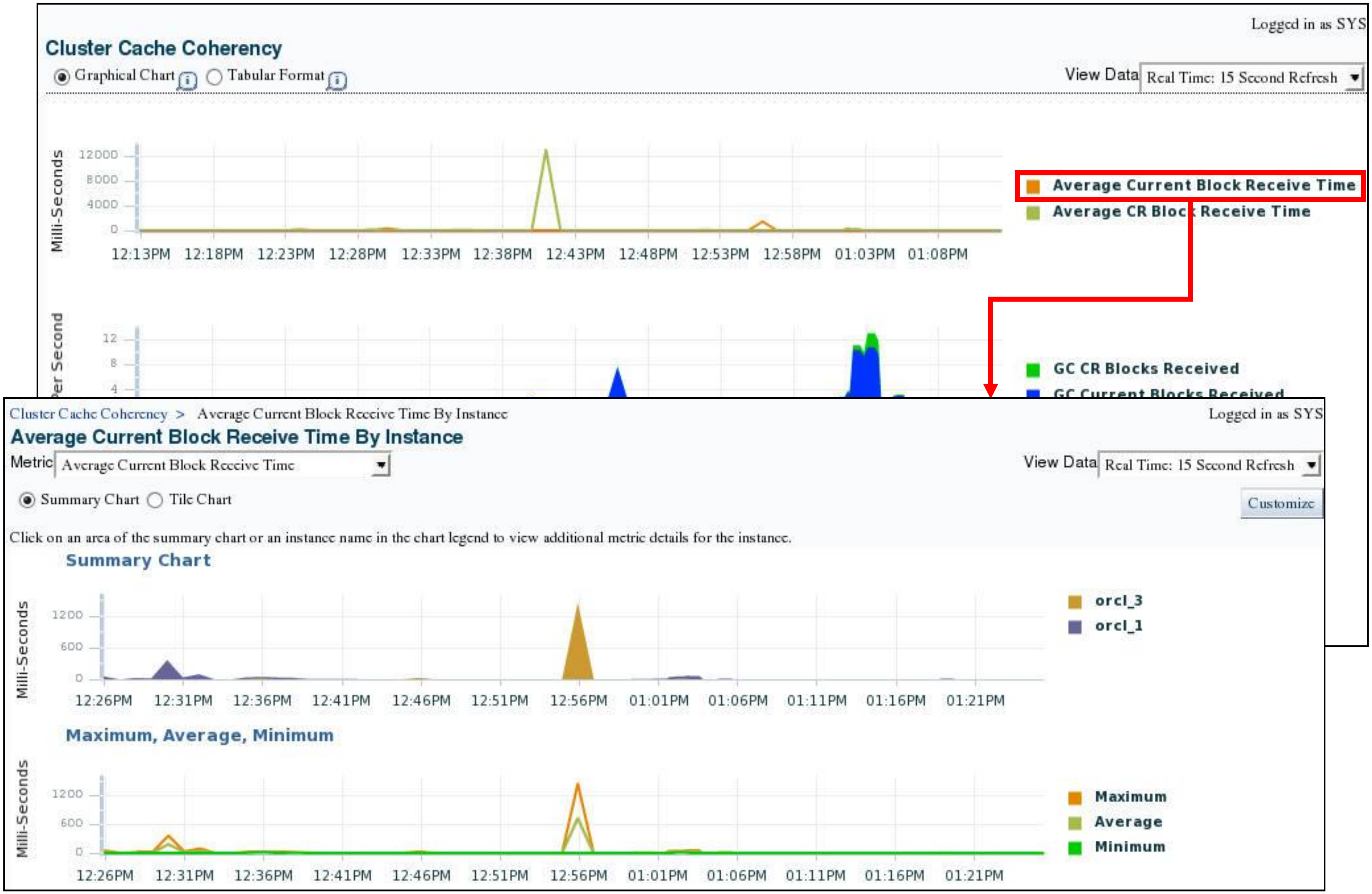
Determining Database Throughput



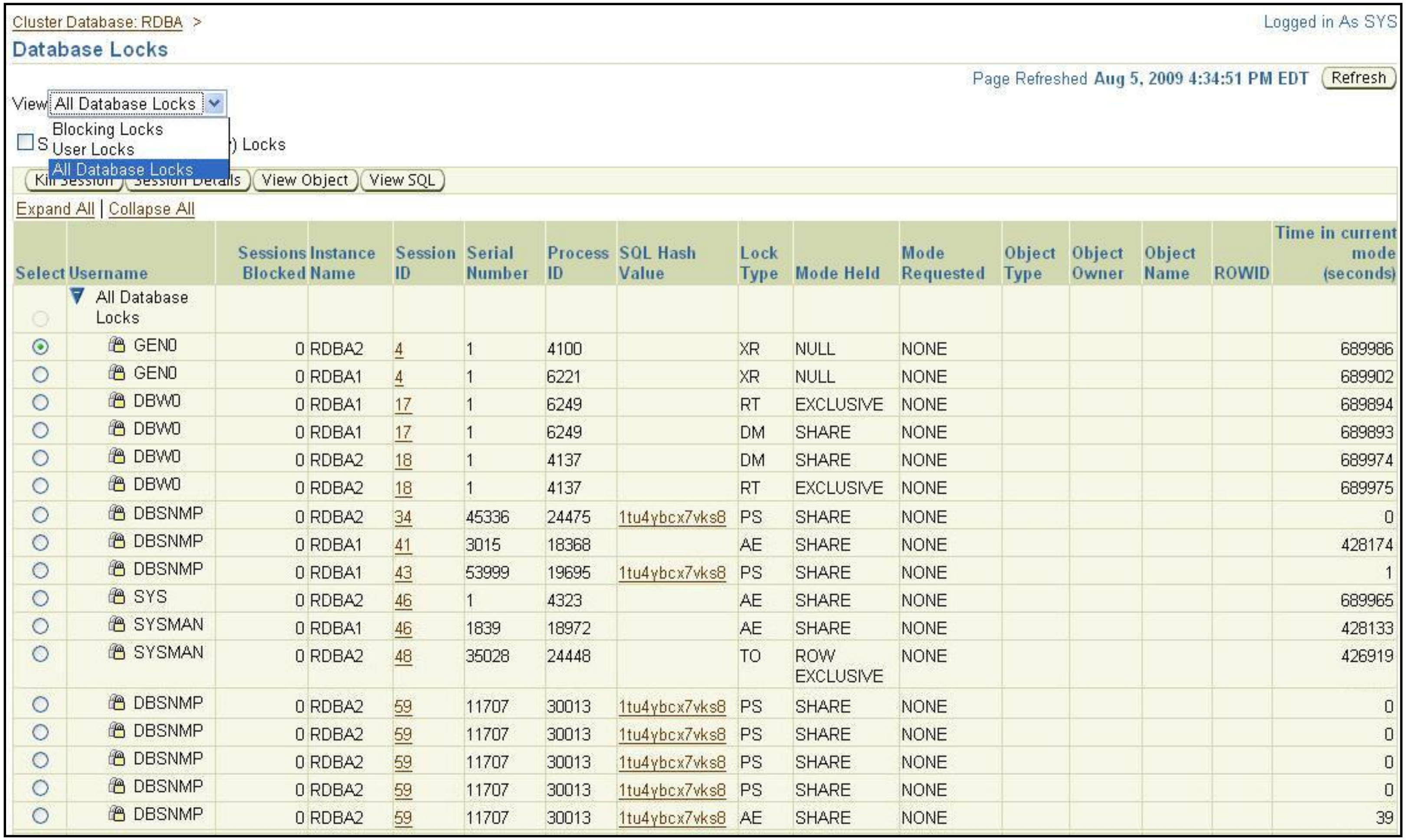
Accessing the Cluster Cache Coherency Page



Accessing the Cluster Cache Coherency Page



Viewing the Database Locks Page



Cluster Database: RDBA >
Logged in As SYS

Database Locks

Page Refreshed **Aug 5, 2009 4:34:51 PM EDT** Refresh

View All Database Locks

Blocking Locks
User Locks
All Database Locks

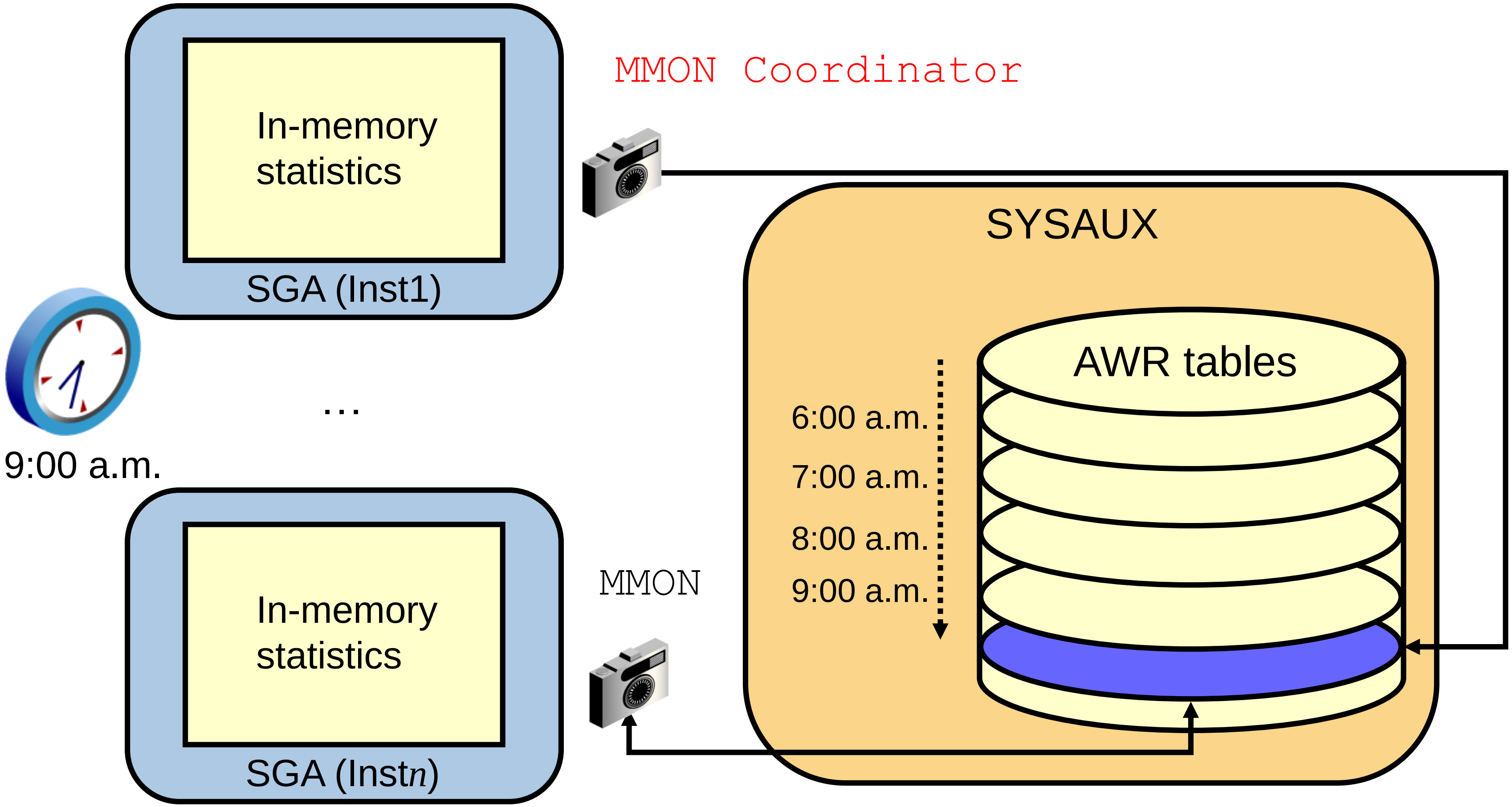
S) Locks

Kill Session | Session Details | View Object | View SQL

Expand All | Collapse All

Select	Username	Sessions Blocked	Instance Name	Session ID	Serial Number	Process ID	SQL Hash Value	Lock Type	Mode Held	Mode Requested	Object Type	Object Owner	Object Name	ROWID	Time in current mode (seconds)
	All Database Locks														
◉	GEN0	0	RDBA2	4	1	4100		XR	NULL	NONE					689986
○	GEN0	0	RDBA1	4	1	6221		XR	NULL	NONE					689902
○	DBW0	0	RDBA1	17	1	6249		RT	EXCLUSIVE	NONE					689894
○	DBW0	0	RDBA1	17	1	6249		DM	SHARE	NONE					689893
○	DBW0	0	RDBA2	18	1	4137		DM	SHARE	NONE					689974
○	DBW0	0	RDBA2	18	1	4137		RT	EXCLUSIVE	NONE					689975
○	DBSNMP	0	RDBA2	34	45336	24475	1tu4ybcx7vks8	PS	SHARE	NONE					0
○	DBSNMP	0	RDBA1	41	3015	18368		AE	SHARE	NONE					428174
○	DBSNMP	0	RDBA1	43	53999	19695	1tu4ybcx7vks8	PS	SHARE	NONE					1
○	SYS	0	RDBA2	46	1	4323		AE	SHARE	NONE					689965
○	SYSMAN	0	RDBA1	46	1839	18972		AE	SHARE	NONE					428133
○	SYSMAN	0	RDBA2	48	35028	24448		TO	ROW EXCLUSIVE	NONE					426919
○	DBSNMP	0	RDBA2	59	11707	30013	1tu4ybcx7vks8	PS	SHARE	NONE					0
○	DBSNMP	0	RDBA2	59	11707	30013	1tu4ybcx7vks8	PS	SHARE	NONE					0
○	DBSNMP	0	RDBA2	59	11707	30013	1tu4ybcx7vks8	PS	SHARE	NONE					0
○	DBSNMP	0	RDBA2	59	11707	30013	1tu4ybcx7vks8	PS	SHARE	NONE					0
○	DBSNMP	0	RDBA2	59	11707	30013	1tu4ybcx7vks8	AE	SHARE	NONE					39

AWR Snapshots in RAC



AWR Reports and RAC: Overview

WORKLOAD REPOSITORY REPORT (RAC)

Database Summary

Database				Snapshot Ids		Number of Instances		Number of Hosts		Report Total (minutes)	
Id	Name	RAC	Block Size	Begin	End	In Report	Total	In Report	Total	DB time	Elapsed time
1352492209	ORCL	YES	8192	108	133	2	2	2	2	81.45	1,512.36

Database Instances Included In Report

- Listed in order of instance number, I#

I#	Instance	Host	Startup	Begin Snap Time	End Snap Time	Release	Elapsed Time(min)	DB time(min)	Up Time(hrs)	Avg Active Sessions	Platform
1	orcl_1	host02.example.com	29-Aug-13 09:15	02-Sep-13 18:48	03-Sep-13 19:46	12.1.0.1.0	1,498.33	18.75	130.52	0.01	Linux x86 64-bit
3	orcl_3	host01.example.com	29-Aug-13 06:55	02-Sep-13 19:00	03-Sep-13 20:00	12.1.0.1.0	1,500.65	62.70	133.09	0.04	Linux x86 64-bit

Main Report

- Report Summary
- OS Statistics
- Time Model Statistics
- Wait Events Statistics
- Global Activity Load Profile
- Global Cache and Enqueue Statistics Summary
- Global CR Served Stats
- Global CURRENT Served Stats
- Global Cache Transfer Stats
- Interconnect Stats
- SQL Statistics
- Global Activity Statistics
- I/O Statistics
- Library Cache Statistics
- Memory Statistics
- Supplemental Information
- ADDM Reports

Report Summary

Top ADDM Findings by Average Active Sessions

Finding Name	Avg active sessions of the task	Percent active sessions of finding	Task Name	Begin Snap Time	End Snap Time
Hard Parse	.67	41.97	ADDM:1352492209_112	02-Sep-13 21:48	02-Sep-13 23:00
Top SQL Statements	.67	34.00	ADDM:1352492209_112	02-Sep-13 21:48	02-Sep-13 23:00
"Scheduler" Wait Class	.67	17.54	ADDM:1352492209_112	02-Sep-13 21:48	02-Sep-13 23:00
Undersized SGA	.67	19.47	ADDM:1352492209_112	02-Sep-13 21:48	02-Sep-13 23:00
Hard Parse Due to Parse Errors	.67	18.55	ADDM:1352492209_112	02-Sep-13 21:48	02-Sep-13 23:00

Cache Sizes

- All values are in Megabytes
- Listed in order of instance number, I#
- End values displayed only if different from Begin values

	Memory Target		Sga Target		DB Cache		Shared Pool		Large Pool		Java Pool		Streams Pool		PGA Target		
I#	Begin	End	Begin	End	Begin	End	Begin	End	Begin	End	Begin	End	Begin	End	Begin	End	Log Buffer
1			976		408	360	504	552	4		4				328		5.18
3			976		336		576		4		4				328		5.18
Avg			976		372	360	540	552	4		4				328		5.18
Min			976		336	360	504	552	4		4				328		5.18
Max			976		408	360	576	552	4		4				328		5.18

Global Cache and Enqueue Statistics Summary

- Global Cache Efficiency Percentages
- Global Cache and Enqueue Workload Characteristics
- Global Cache and Enqueue Messaging Statistics

Back to Top

Global Cache Efficiency Percentages

	Buffer Access		
I#	Local %	Remote %	Disk %
1	99.23	0.16	0.61
3	99.23	0.19	0.58

Back to Global Cache and Enqueue Statistics Summary
Back to Top

Global Cache and Enqueue Workload Characteristics

		CR Blocks					CU Blocks			
I#	GE Get Time (ms)	Receive Time (ms)	Build Time (ms)	Send Time (ms)	Flush Time (ms)	Log Flush CR Srvd %	Receive Time (ms)	Pin Time (ms)	Send Time (ms)	Flush Time (ms)
1	0.28	1.67	0.00	0.00	17.89	6.20	3.81	0.05	0.00	19.00
3	0.52	10.50	0.00	0.00	12.03	2.05	5.69	0.01	0.00	14.24

Back to Global Cache and Enqueue Statistics Summary
Back to Top

Global Cache and Enqueue Messaging Statistics

	Queue Time (ms)			Process Time		% Messages Sent		
I#	Sent	on ksxp	Received	GCS msgs	GES msgs	Direct	Indirect	Flow Ctrl
1	0.11	1.52	0.03	0.18	0.04	79.18	20.52	0.30
3	0.26	2.77	0.02	0.05	0.03	69.76	27.83	2.41

Active Session History Reports for RAC

- Active Session History (ASH) report statistics provide details about the RAC Database session activity.
- The database records information about active sessions for all active RAC instances.
- Two ASH report sections specific to Oracle RAC are **Top Cluster Events** and **Top Remote Instance**.

ASH Report For ORCL/orcl_3

DB Name	DB Id	Instance	Inst num	Release	RAC	Host
ORCL	1352492209	orcl_3	3	12.1.0.1.0	YES	host01.example.com

CPUs	SGA Size	Buffer Cache	Shared Pool	ASH Buffer Size
1	972M (100%)	524M (53.9%)	384M (39.5%)	2.0M (0.2%)

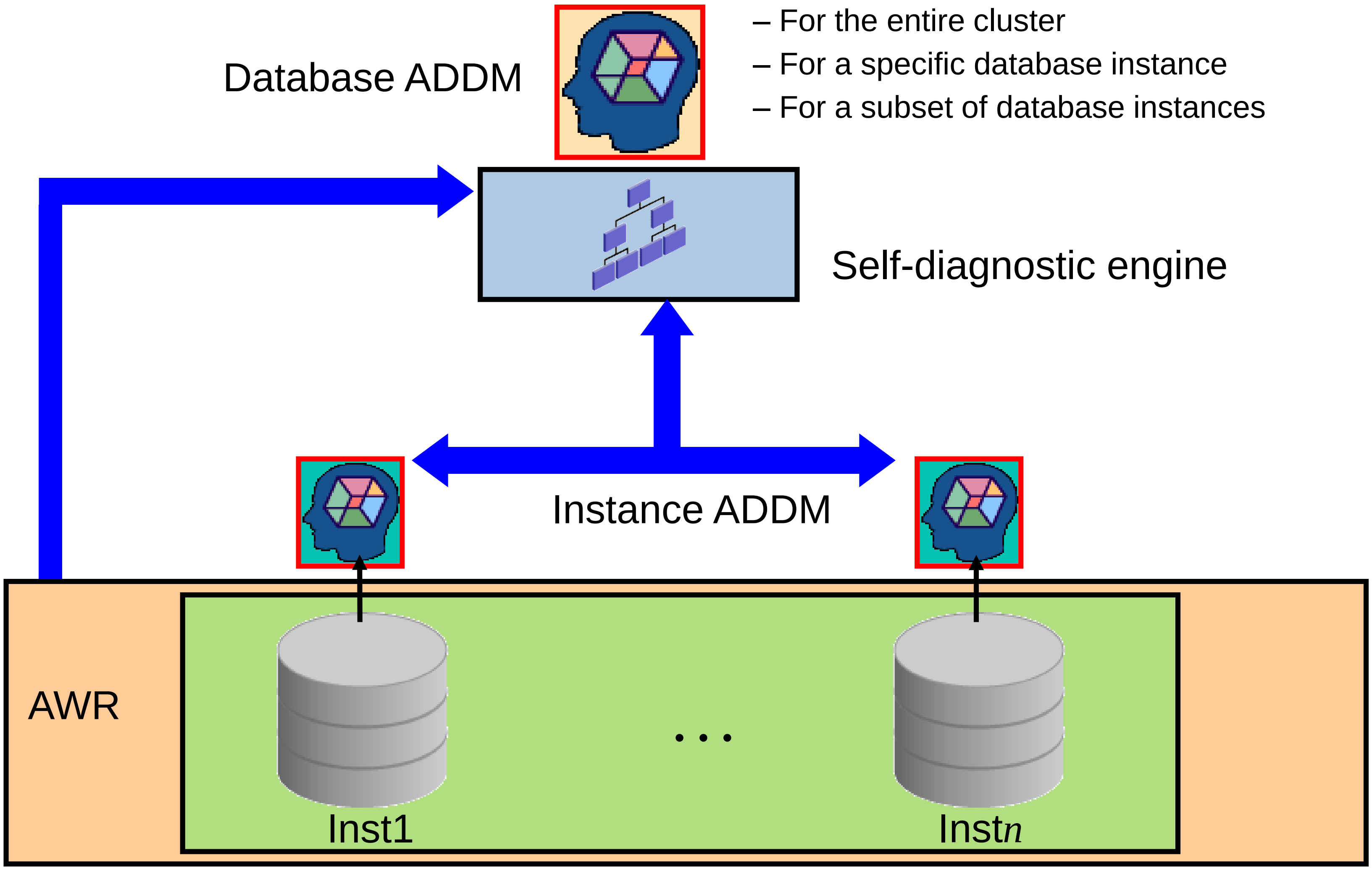
	Sample Time	Data Source
Analysis Begin Time:	05-Sep-13 05:17:22	V$ACTIVE_SESSION_HISTORY
Analysis End Time:	05-Sep-13 05:32:28	V$ACTIVE_SESSION_HISTORY
Elapsed Time:	15.1 (mins)	
Sample Count:	225	
Average Active Sessions:	0.25	
Avg. Active Session per CPU:	0.25	
Report Target:	None specified	

ASH Report

- Top Events
- Load Profile
- Top SQL
- Top PL/SQL
- Top Java
- Top Call Types
- Top Sessions
- Top Objects/Files/Latches
- Activity Over Time

Automatic Database Diagnostic Monitor for RAC

ADDM can perform analysis:
– For the entire cluster
– For a specific database instance
– For a subset of database instances



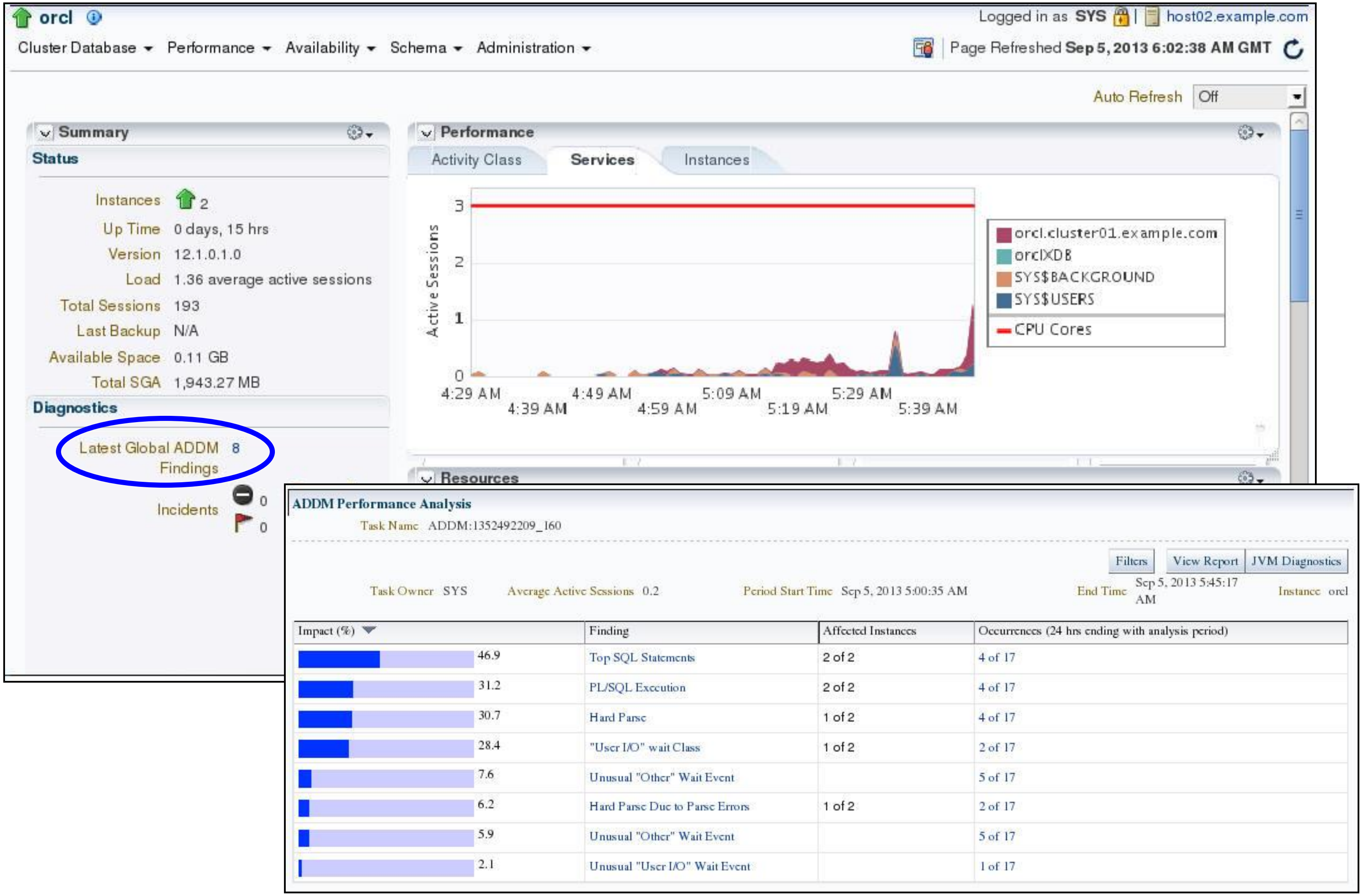
Automatic Database Diagnostic Monitor for RAC

- Identifies the most critical performance problems for the entire RAC cluster database
- Runs automatically when taking AWR snapshots
- Performs database-wide analysis of:
  - Global resources (for example I/O and global locks)
  - High-load SQL and hot blocks
  - Global cache interconnect traffic
  - Network latency issues
  - Skew in instance response times
- Is used by DBAs to analyze cluster performance
- Does not require investigation of *n* reports to spot common problems

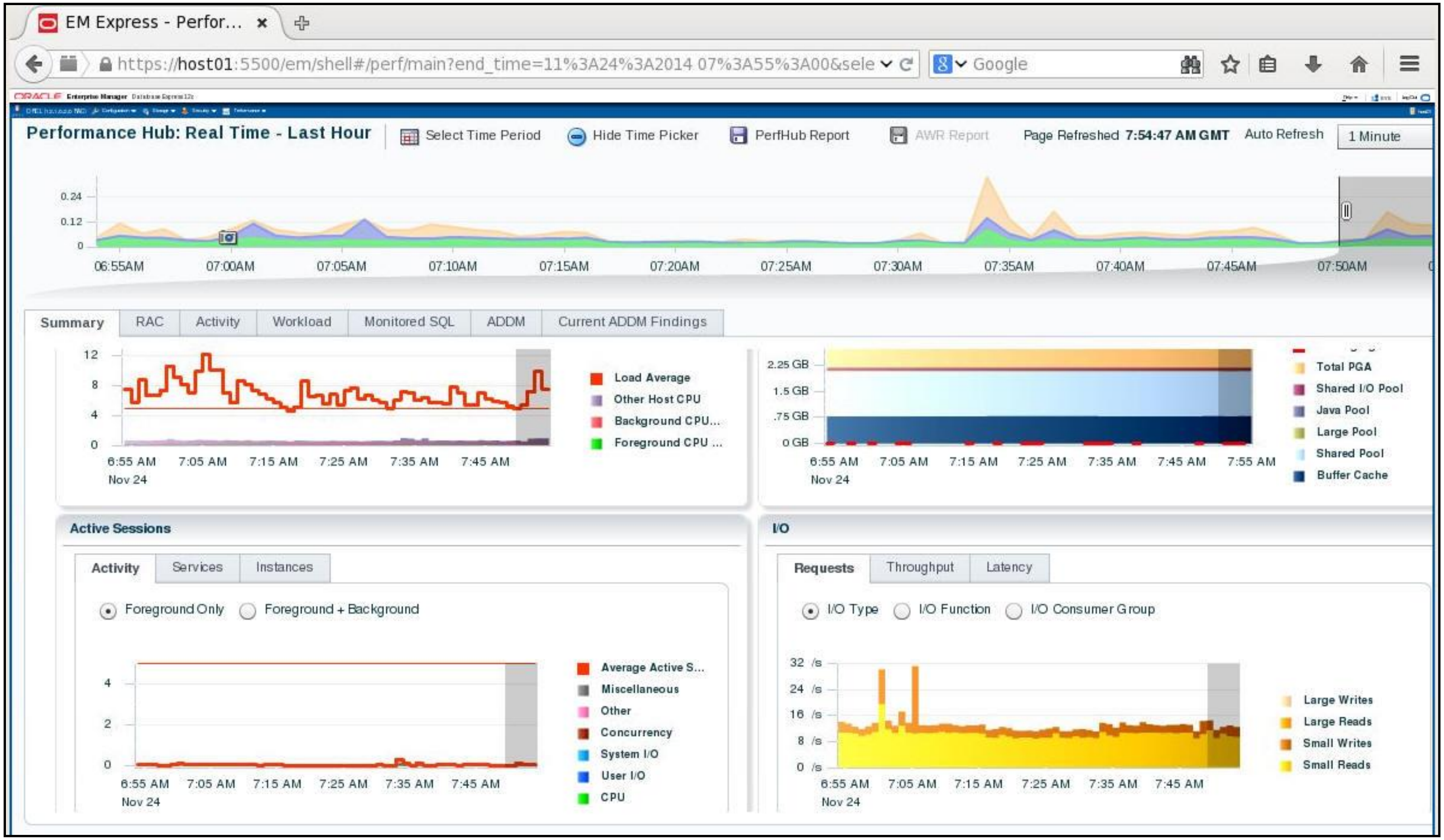
What Does ADDM Diagnose for RAC?

- Latency problems in interconnect
- Congestion (identifying top instances affecting the entire cluster)
- Contention (buffer busy, top objects, and so on)
- Top consumers of multiblock requests
- Lost blocks
- Reports information about interconnect devices; warns about using PUBLIC interfaces
- Reports throughput of devices, and how much of it is used by Oracle and for what purpose (GC, locks, PQ)

EM Support for ADDM for RAC



EM Database Express Performance Hub



Quiz

Although there are specific tuning areas for RAC, such as instance recovery and interconnect traffic, you get most benefits by tuning your system like a single-instance system.

a. True

b. False

Quiz

Which of the following RAC tuning tips are correct?

a. Application tuning is often the most beneficial.
b. Reduce long full-table scans in OLTP systems.
c. Eliminate sequence caches.
d. Use partitioning to reduce inter-instance traffic.
e. Configure the interconnects properly.

Summary

In this lesson, you should have learned how to:

- Determine RAC-specific tuning components
- Determine RAC-specific wait events, global enqueues, and system statistics
- Implement the most common RAC tuning tips
- Use the Cluster Database Performance pages
- Use the Automatic Workload Repository (AWR) in RAC
- Use Automatic Database Diagnostic Monitor (ADDM) in RAC

Practice 8: Overview

This practice covers manually discovering performance issues by using the EM performance pages as well as ADDM.

Managing High Availability of Services

Objectives

After completing this lesson, you should be able to:

- Configure and manage services in a RAC environment
- Use services with client applications
- Use services with the Database Resource Manager
- Use services with the Scheduler
- Configure services aggregation and tracing

Oracle Services

- To manage workloads or a group of applications, you can define services for a particular application or a subset of an application's operations.
- You can also group work by type under services.
- For example, OLTP users can use one service while batch processing can use another to connect to the database.
- Users who share a service should have the same service-level requirements.
- Use **`srvctl`** or Enterprise Manager to manage services, ***not*** **`DBMS_SERVICE`**.

Service Usage in an Oracle RAC Database

- Services provide location transparency.
- A service name can identify multiple database instances and an instance can belong to multiple services.
- Resource profiles are automatically created when you define a service.
  - A resource profile describes how Oracle Clusterware should manage the service.
  - Resource profiles also define service dependencies for the instance and the database.
- Services are integrated with Resource Manager, enabling you to restrict the resources that users use to connect to an instance by using a service.

Parallel Operations and Services

- By default, in an RAC environment, a SQL statement run in parallel can run across all of the nodes in the cluster.
- To perform well, the interconnect must be sized correctly as inter-node parallel execution may result in a lot of interconnect traffic.
- You can control parallel execution in a RAC environment with the `PARALLEL_FORCE_LOCAL` initialization parameter.
  - The parallel execution servers can only execute on the same Oracle RAC node where the SQL statement was started.
- Services can be used to limit the number of instances that participate in a parallel SQL operation.

Service Characteristics

The characteristics of a service include:

- Service Name
- Service Edition
- Service Management Policy
- Database Role for a Service
- Instance Preference
- Server Pool Assignment
- Load Balancing Advisory Goal for Run-time Connection Load Balancing
- Connection Load Balancing Goal

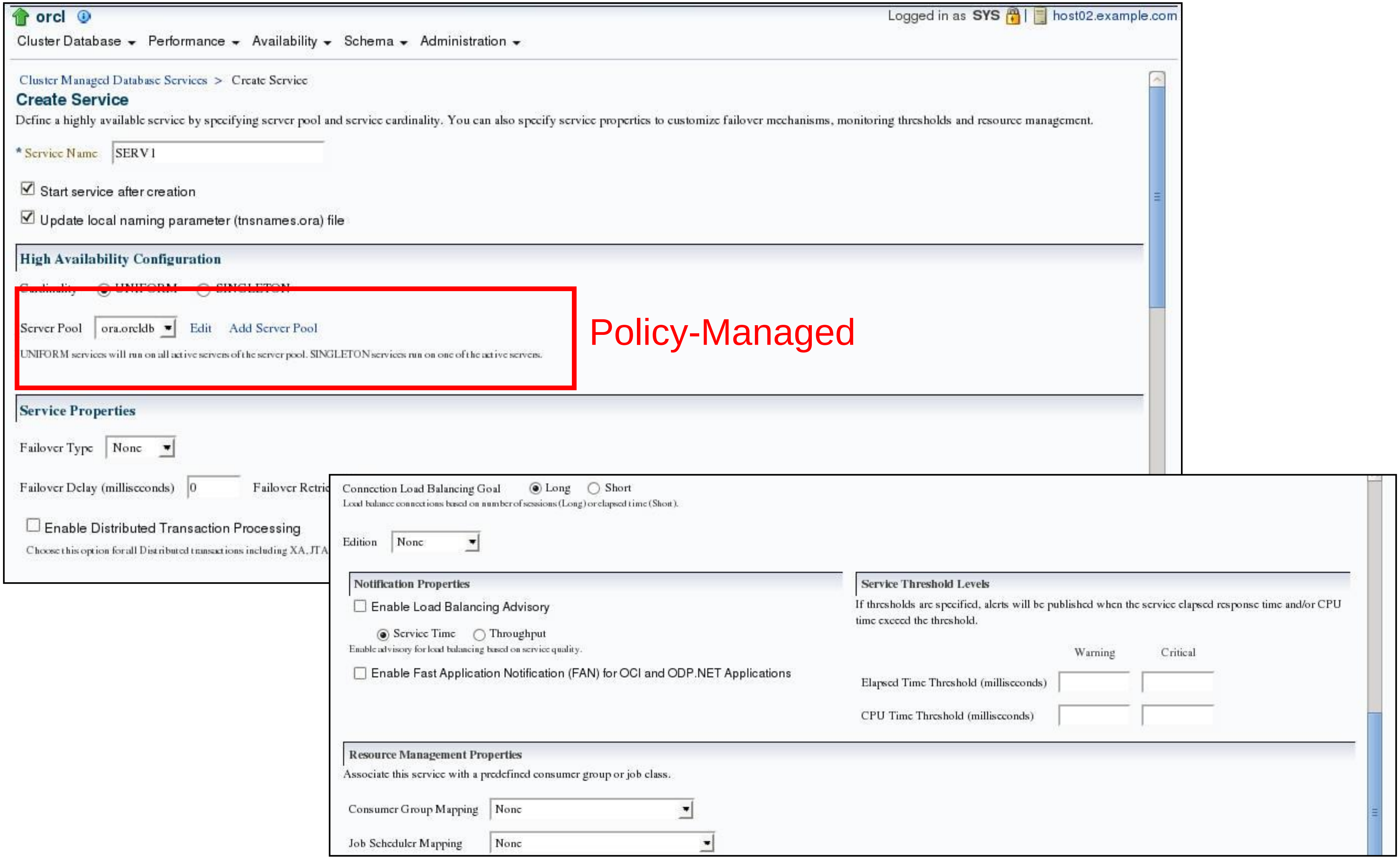
Default Service Connections

- Your RAC database includes an Oracle database service identified by `DB_UNIQUE_NAME`, if set.
  - `DB_NAME` or `PDB_NAME`, if not.
- This default service is always available on all instances in an Oracle RAC environment.
- Additionally, the database supports two internal services:
  - `SYS$BACKGROUND` is used by background processes only.
  - `SYS$USERS` is the default service for user sessions that are not associated with any application service.

Restricted Service Registration

- Restricted Service Registration allows listener registration only from local IP addresses, by default.
  - Provides the ability to configure and update a set of addresses or subnets from which registration requests are allowed by the listener.
- Database Instance registration with a listener succeeds only when the request originates from a valid node.
- The network administrator can specify a list of valid nodes, excluded nodes, or disable valid node checking.
- The control of dynamic registration results in increased manageability and security of Oracle RAC deployments.
- Valid node checking for registration (VNCR) is enabled by default.

Creating Service with Enterprise Manager



Creating Services with `SRVCTL`

- To create a service called `GL` with preferred instance RAC02 and an available instance RAC01:

```
$ srvctl add service -db PROD1 -service GL -preferred RAC02
-available RAC01
```

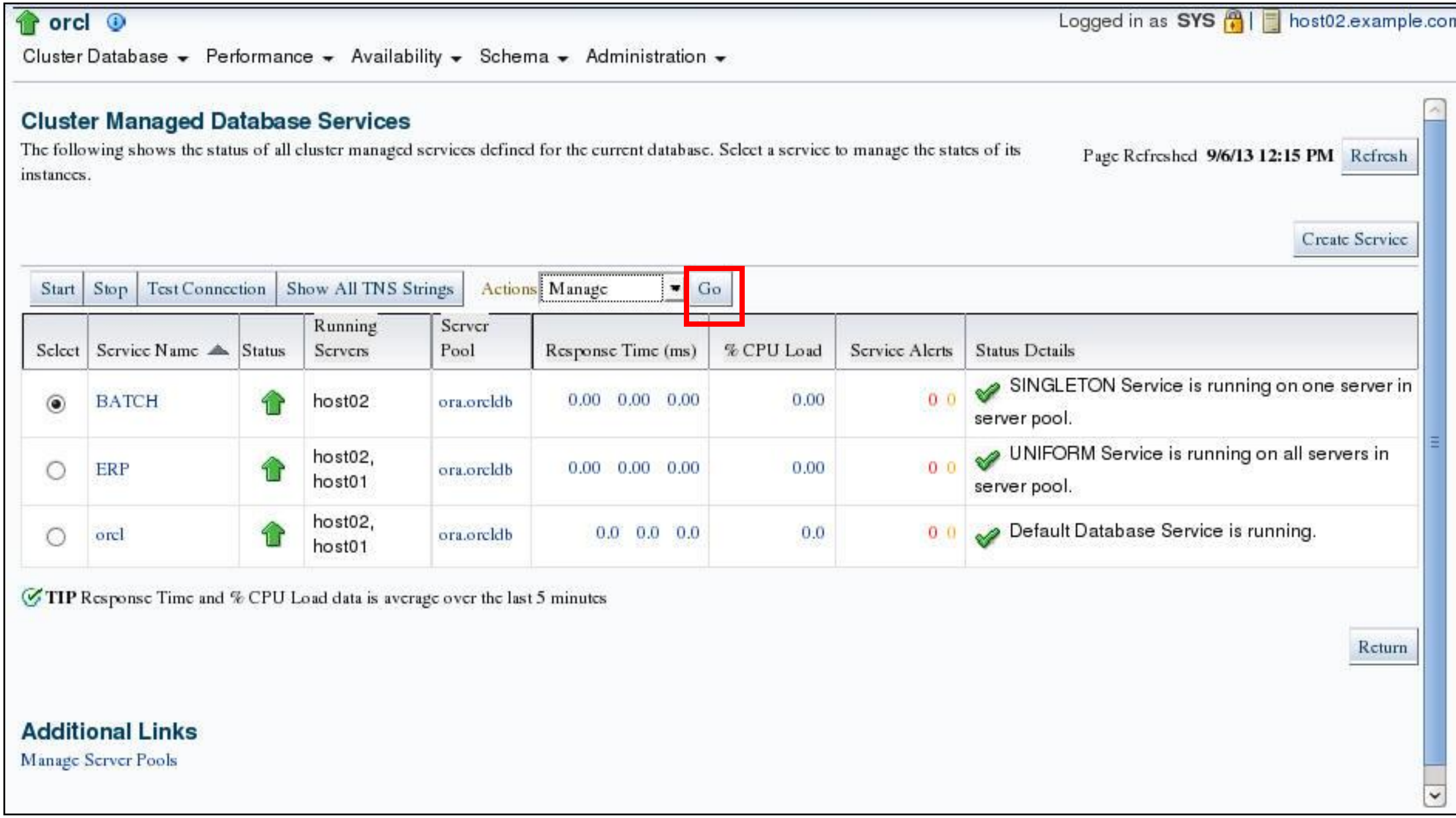
- To create a service called `AP` with preferred instance RAC01 and an available instance RAC02:

```
$ srvctl add service -db PROD1 -service AP -preferred RAC01
-available RAC02
```

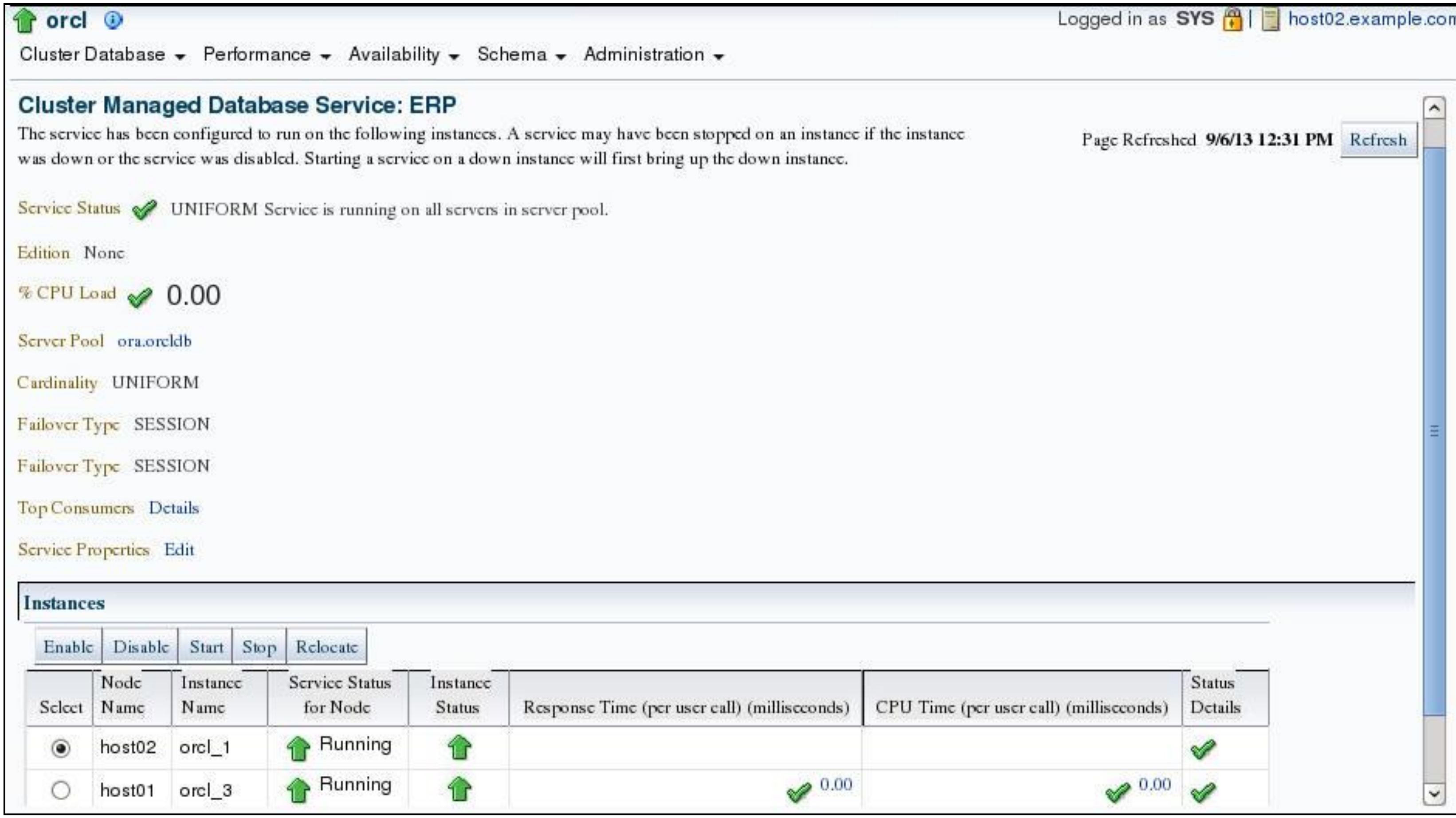
- Create a `SINGLETON` service called `BATCH` using server pool `SP1` and a `UNIFORM` service called `ERP` using pool `SP2`:

```
$ srvctl add service -db PROD2 -service BATCH -serverpool
SP1 -cardinality singleton -policy manual

$ srvctl add service -db PROD2 -service ERP -serverpool
SP2 -cardinality UNIFORM -policy manual
```

Managing Services with Enterprise Manager



Managing Services with EM



Managing Services with `srvctl`

- Start a named service on all configured instances:

```
$ srvctl start service -db orcl -service AP
```

- Stop a service:

```
$ srvctl stop service -db orcl -service AP -instance orcl4
```

- Disable a service at a named instance:

```
$ srvctl disable service -db orcl -service AP -instance orcl4
```

- Set an available instance as a preferred instance:

```
$ srvctl modify service -db orcl -service AP -instance orcl5
-preferred
```

- Relocate a service from one instance to another:

```
$ srvctl relocate service -db orcl -service AP -oldinst
orcl5 -newinst orcl4
```

Using Services with Client Applications

```
ERP=(DESCRIPTION=        ## Using SCAN  ##
 (LOAD_BALANCE=on)
      (ADDRESS=(PROTOCOL=TCP)(HOST=cluster01-scan)(PORT=1521))
     (CONNECT_DATA=(SERVICE_NAME=ERP)))
```

```
ERP=(DESCRIPTION=        ## Using VIPs ##
   (LOAD_BALANCE=on)
     (ADDRESS=(PROTOCOL=TCP)(HOST=node1-vip)(PORT=1521))
     (ADDRESS=(PROTOCOL=TCP)(HOST=node2-vip)(PORT=1521))
     (ADDRESS=(PROTOCOL=TCP)(HOST=node3-vip)(PORT=1521))
    (CONNECT_DATA=(SERVICE_NAME=ERP)))
```

```
url="jdbc:oracle:oci:@ERP"       ## Thick JDBC ##
```

```
url="jdbc:oracle:thin:@(DESCRIPTION=        ## Thin JDBC ##
     (LOAD_BALANCE=on)
       (ADDRESS=(PROTOCOL=TCP)(HOST=cluster01-scan)(PORT=1521)))
     (CONNECT_DATA=(SERVICE_NAME=ERP)))"
```

Services and Connection Load Balancing

- The two load balancing methods that you can implement are:
  - **Client-side load balancing:** Balances the connection requests across the listeners
  - **Server-side load balancing:** The listener directs a connection request to the best instance currently providing the service by using the load balancing advisory (LBA).
- FAN, Fast Connection Failover, and LBA depend on a connection load balancing configuration that includes setting the connection load balancing goal for the service.
- The load balancing goal for the service can be either:
  - **`LONG`**: For applications having long-lived connections. This is typical for connection pools and SQL*Forms sessions.
  - **`SHORT`**: For applications that have short-lived connections

```
srvctl modify service -service service_name -clbgoal
LONG|SHORT
```

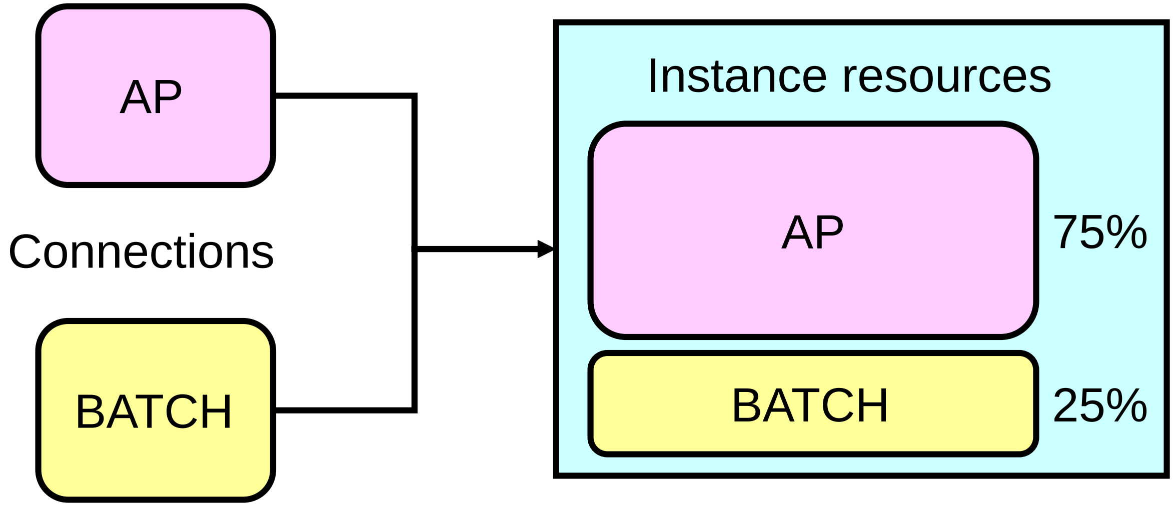
Services and Transparent Application Failover

- Services simplify the deployment of Transparent Application Failover (TAF).
- You can define a TAF policy for a service and all connections using this service will automatically have TAF enabled.
- The TAF setting on a service can be **`NONE`**, **`BASIC`**, or **`PRECONNECT`** and overrides any TAF setting in the client connection definition.
- To define a TAF policy for a service, the **`srvctl`** utility can be used as follows:

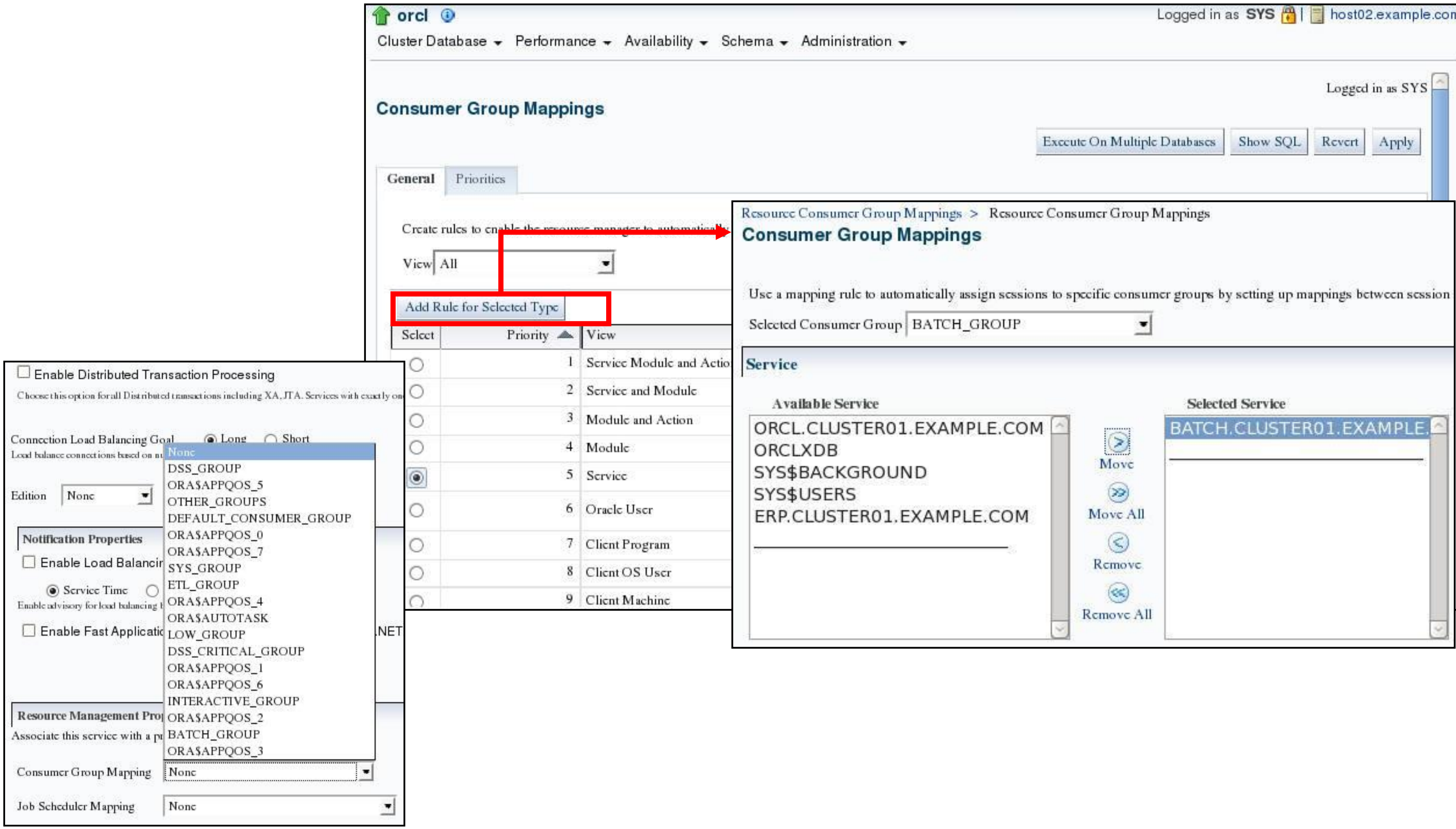
```
srvctl modify service -db crm -service GL
-failovermethod BASIC -failovertype SELECT
-failoverretry 10 -failoverdelay 30
```

Using Services with the Resource Manager

- Consumer groups are automatically assigned to sessions based on session services.
- Work is prioritized by service inside one instance.

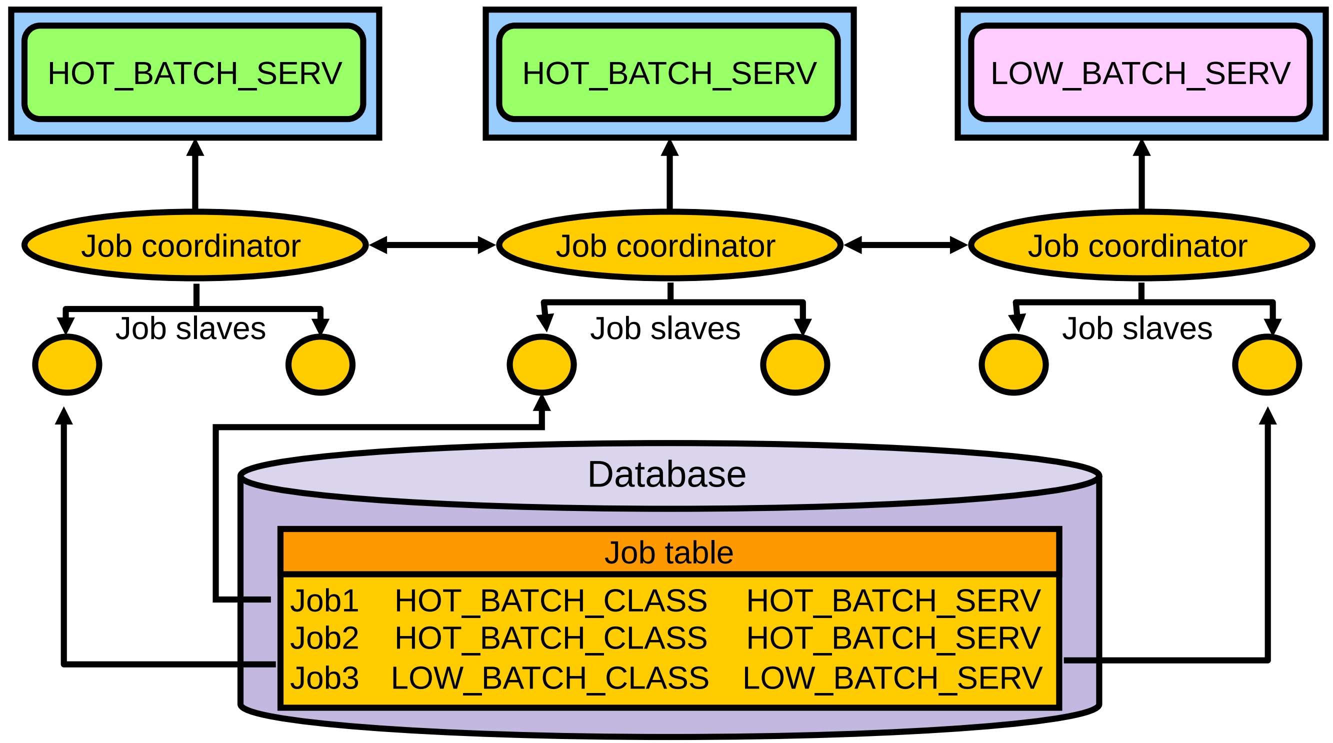


Services and Resource Manager with EM

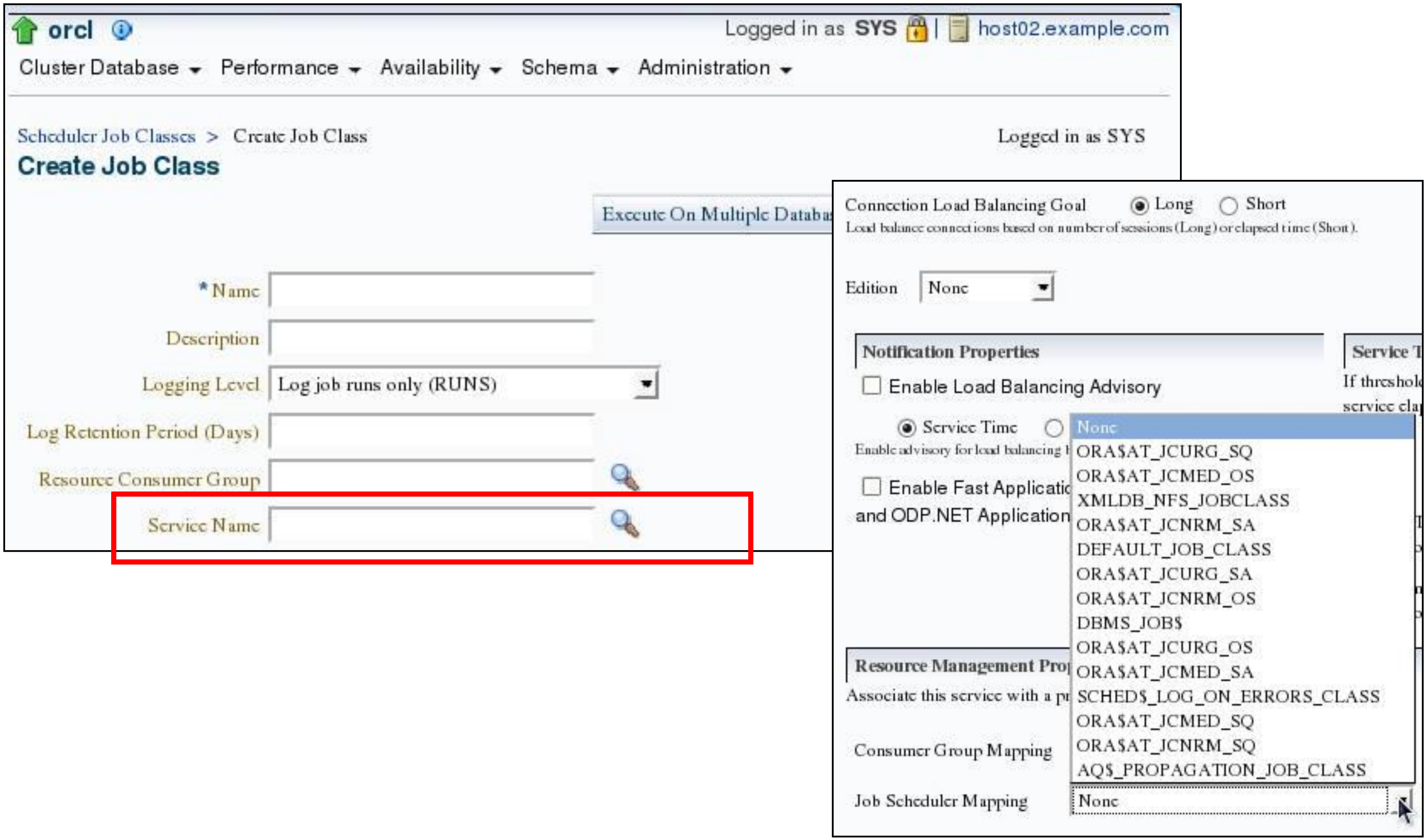


Using Services with the Scheduler

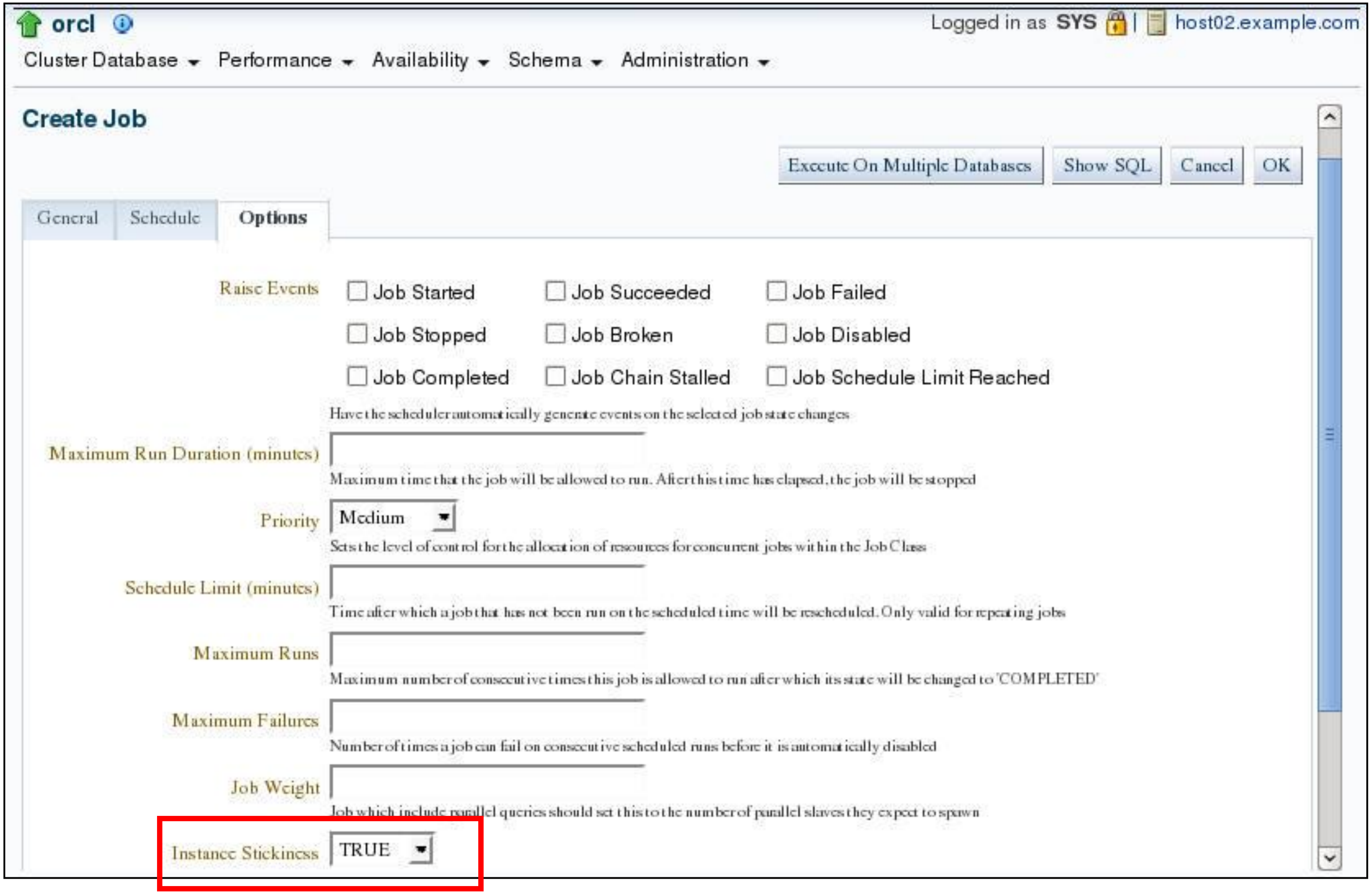
- Services are associated with Scheduler classes.
- Scheduler jobs have service affinity:
  - High availability
  - Load balancing



Services and the Scheduler with EM



Services and the Scheduler with EM



Using Distributed Transactions with RAC

- An XA transaction can span RAC instances, allowing any application that uses XA to take full advantage of the Oracle RAC environment.
- Tightly coupled XA transactions no longer require the special type of singleton services (DTP).
- XA transactions are transparently supported on Oracle RAC databases with any type of services configuration.
- However, DTP services will improve performance for many distributed transaction scenarios.
- DTP services allow you to direct all branches of a distributed transaction to a single instance in the cluster.
- To load balance, it is better to have several groups of smaller application servers with each group directing its transactions to a single service.

Distributed Transactions and Services

To create distributed transaction processing (DTP) services for distributed transaction processing, perform the following steps:

1. Create a singleton service using EM or SRVCTL. For an administrator-managed database, define only one instance as the preferred instance:

```
$ srvctl add service -db crm -service xa_01.example.com
-preferred RAC01 -available RAC02,RAC03
```

   For a policy-managed database, specify the server pool to use, and set the cardinality of the service to `SINGLETON`:

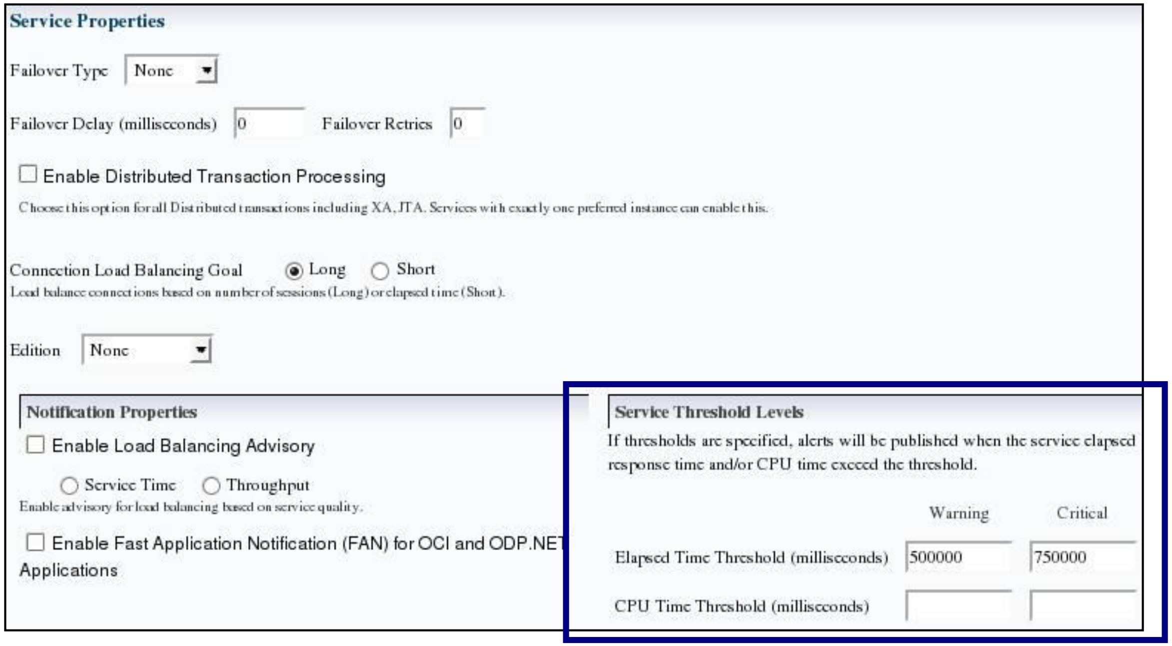
```
$ srvctl add service -db crm -service xa_01.example.com
-serverpool dtp_pool -cardinality SINGLETON
```

2. Set the DTP parameter (`-dtp`) for the service to `TRUE`:

```
$ srvctl modify service -db crm -service xa_01.example.com
-dtp TRUE
```

Service Thresholds and Alerts

- Service-level thresholds enable you to compare achieved service levels against accepted minimum required levels.
- You can explicitly specify two performance thresholds for each service:
  - **`SERVICE_ELAPSED_TIME`**: The response time for calls
  - **`SERVICE_CPU_TIME`**: The CPU time for calls



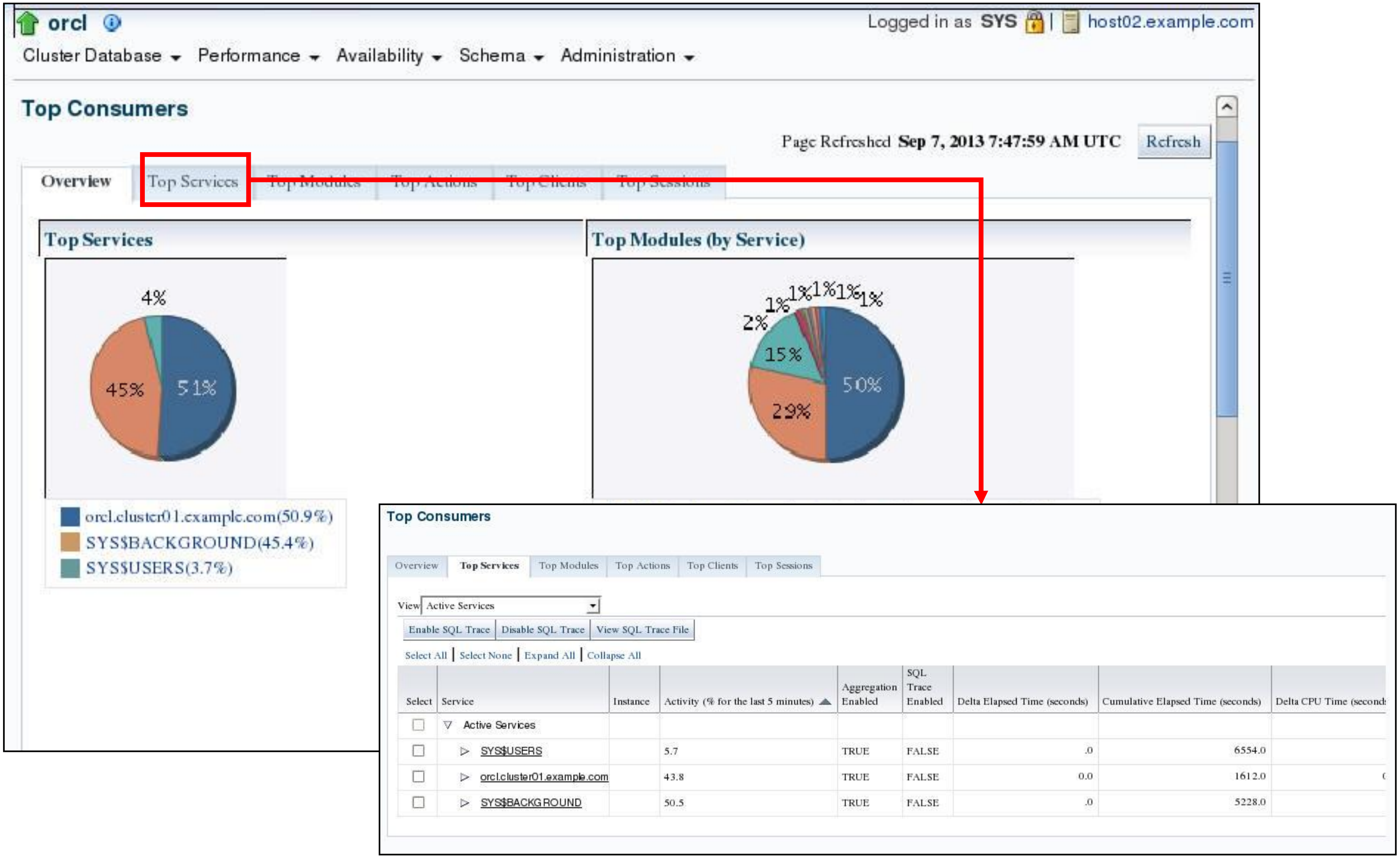
Services and Thresholds Alerts: Example

```
EXECUTE DBMS_SERVER_ALERT.SET_THRESHOLD(
METRICS_ID => DBMS_SERVER_ALERT.ELAPSED_TIME_PER_CALL
, warning_operator => DBMS_SERVER_ALERT.OPERATOR_GE ,
  warning_value => '500000' , critical_operator =>
  DBMS_SERVER_ALERT.OPERATOR_GE
, critical_value => '750000'
, observation_period => 30
, consecutive_occurrences => 5
, instance_name => NULL
, object_type => DBMS_SERVER_ALERT.OBJECT_TYPE_SERVICE
, object_name => 'payroll');
```

Service Aggregation and Tracing

- Statistics are always aggregated by service to measure workloads for performance tuning.
- Statistics can be aggregated at finer levels:
  - MODULE
  - ACTION
  - Combination of SERVICE_NAME, MODULE, ACTION
- Tracing can be done at various levels:
  - SERVICE_NAME
  - MODULE
  - ACTION
  - Combination of SERVICE_NAME, MODULE, ACTION
- This is useful for tuning systems that use shared sessions.

Top Services Performance Page



Service Aggregation Configuration

- Automatic service aggregation level of statistics
- `DBMS_MONITOR` used for finer granularity of service aggregations:
  - `SERV_MOD_ACT_STAT_ENABLE`
  - `SERV_MOD_ACT_STAT_DISABLE`
- Possible additional aggregation levels:
  - `SERVICE_NAME`/`MODULE`
  - `SERVICE_NAME`/`MODULE`/`ACTION`
- Tracing services, modules, and actions:
  - `SERV_MOD_ACT_TRACE_ENABLE`
  - `SERV_MOD_ACT_TRACE_DISABLE`
- Database settings persist across instance restarts.

Service, Module, and Action Monitoring

- For the `ERP` service, enable monitoring for the exceptions pay action in the `PAYROLL` module.

```
EXEC DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(
service_name => 'ERP', module_name=> 'PAYROLL',
action_name => 'EXCEPTIONS PAY')
```

- For the `ERP` service, enable monitoring for all the actions in the `PAYROLL` module:

```
EXEC DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name =>
'ERP', module_name=> 'PAYROLL', action_name => NULL);
```

- For the `HOT_BATCH` service, enable monitoring for all actions in the `POSTING` module:

```
EXEC DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name =>
'HOT_BATCH', module_name =>'POSTING', action_name => NULL);
```

Service Performance Views

- Service, module, and action information in:
  - V$SESSION
  - V$ACTIVE_SESSION_HISTORY
- Service performance in:
  - V$SERVICE_STATS
  - V$SERVICE_EVENT
  - V$SERVICE_WAIT_CLASS
  - V$SERVICEMETRIC
  - V$SERVICEMETRIC_HISTORY
  - V$SERV_MOD_ACT_STATS
  - DBA_ENABLED_AGGREGATIONS
  - DBA_ENABLED_TRACES

Quiz

Which of the following statements regarding Oracle Services is *not* correct?

a. You can group work by type under services.
b. Users who share a service should have the same service-level requirements.
c. You use `DBMS_SERVICE` to manage services, not `srvctl` or Enterprise Manager.

Quiz

Is the following statement regarding performance thresholds true or false? The two performance thresholds that can be explicitly set for each service are:

(a) `SERVICE_ELAPSED_TIME`: The response time for calls

(b) `SERVICE_CPU_TIME`: The CPU time for calls

a. True

b. False

Summary

In this lesson, you should have learned how to:

- Configure and manage services in a RAC environment
- Use services with client applications
- Use services with the Database Resource Manager
- Use services with the Scheduler
- Set performance-metric thresholds on services
- Configure services aggregation and tracing

Practice 9: Overview

This practice covers the following topics:

- Creating and managing services

10

High Availability for Connections and Applications

Objectives

After completing this lesson, you should be able to:

- Configure client-side connect-time load balancing
- Configure client-side connect-time failover
- Configure server-side connect-time load balancing
- Configure Transparent Application Failover (TAF)
- Describe the purpose of Transaction Guard and Application Continuity
- Describe the key concepts relating to Application Continuity

Types of Workload Distribution

- Connection balancing is rendered possible by configuring multiple listeners on multiple nodes:
  - Client-side connect-time load balancing
  - Client-side connect-time failover
  - Server-side connect-time load balancing
- Runtime connection load balancing is rendered possible by using connection pools:
  - Work requests automatically balanced across the pool of connections
  - Native feature of Oracle Universal Connection Pool (UCP) for Java, and ODP.NET connection pool

Client-Side Connect-Time Load Balancing

- Client-side load balancing is defined in `tnsnames.ora` by setting the parameter `LOAD_BALANCE=ON`.
  - When set, the Oracle Database randomly selects an address in the address list, and connects to that node's listener.
  - Client connections are balanced across the available SCAN listeners in the cluster.

```
ERP =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (LOAD_BALANCE=ON)
      (ADDRESS=(PROTOCOL=TCP)(HOST=node1vip)(PORT=1521))
      (ADDRESS=(PROTOCOL=TCP)(HOST=node2vip)(PORT=1521))
    )
    (CONNECT_DATA=(SERVICE_NAME=ERP)))
```

- If SCAN is configured, then client-side load balancing is not relevant for clients supporting SCAN access.
- If SCAN is used, connections are balanced across the three IP addresses defined for SCAN, unless you are using EZConnect.

Fast Application Notification (FAN): Overview

- FAN is used by RAC to notify other processes about configuration and service-level information.
  - This includes service status changes like `UP` or `DOWN` events.
  - `UP` and `DOWN` events can apply to instances, services, and nodes.
- Oracle client drivers and Oracle connection pools respond to FAN events and take immediate action.
- Oracle connection pools use FAN to:
  - Quickly detect failures
  - Balance connections following failures
  - Balance connections after failed components are repaired
- Using FAN events eliminates:
  - Applications waiting on TCP timeouts
  - Time wasted processing the last result at the client after a failure has occurred

Fast Application Notification: Benefits

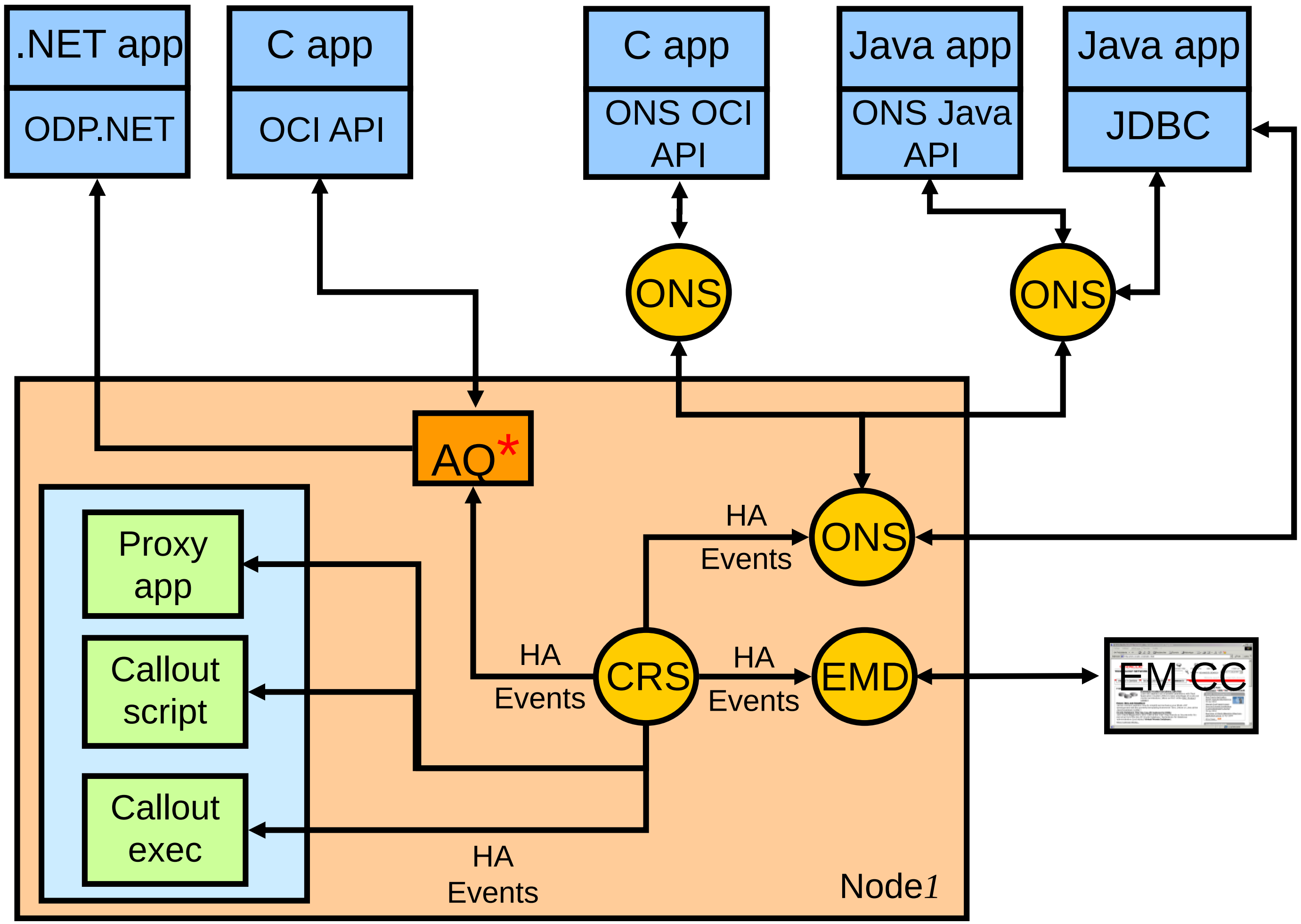
- No need for connections to rely on connection timeouts
- Used by Load Balancing Advisory to propagate load information
- Designed for enterprise application and management console integration
- Reliable distributed system that:
  – Detects high-availability event occurrences in a timely manner
  – Pushes notification directly to your applications
- Tightly integrated with:
  – Oracle JDBC applications using connection pools
  – ODP.NET connection pool, OCI session pool
  – Oracle WebLogic Server Active Gridlink for Oracle RAC

Implementing FAN Events

You can take advantage of FAN events in the following ways:

- Your application can use FAN without programmatic changes if you use an integrated Oracle client:
  - Oracle JDBC Universal Connection Pool (UCP)
  - ODP.NET connection pool
  - Oracle WebLogic Server Active Gridlink for RAC
  - OCI and ODP.NETclients
- Applications can use FAN programmatically:
  - By using the JDBC and RAC FAN API
  - By using callbacks with OCI and ODP.NET to subscribe to FAN events
- You can implement FAN with server-side callouts on your database tier.

FAN and Oracle Integrated Clients



* *Streams is continued for backward compatibility to previous Oracle Database releases.*

FAN-Supported Event Types

Event type	Description
SERVICE	Primary application service
SRV_PRECONNECT	Shadow application service event (mid-tiers and TAF using primary and secondary instances)
SERVICEMEMBER	Application service on a specific instance
DATABASE	Oracle database
INSTANCE	Oracle instance
ASM	Oracle ASM instance
NODE	Oracle cluster node
SERVICEMETRICS	Load Balancing Advisory

FAN Event Reasons

Event Reason	Description
`USER`	User-initiated commands, such as `srvctl` and `sqlplus`
`FAILURE`	Managed resource polling checks detecting a failure
`DEPENDENCY`	Dependency of another managed resource that triggered a failure condition
`UNKNOWN`	Unknown or internal application state when an event is triggered
`AUTOSTART`	Initial cluster boot: Managed resource has profile attribute `AUTO_START=1`, and was offline before the last Oracle Clusterware shutdown.
`BOOT`	Initial cluster boot: Managed resource was running before the last Oracle Clusterware shutdown.
`PUBLIC_NW_DOWN`	The node is up, but a downed network prevents connectivity.
`MEMBER_LEAVE`	A node has failed and is no longer part of the cluster.

FAN Event Status

Event status	Description
`UP`	Managed resource comes up.
`DOWN`	Managed resource goes down.
`PRECONN_UP`	Shadow application service comes up.
`PRECONN_DOWN`	Shadow application service goes down.
`NODEDOWN`	Managed node goes down.
`NOT_RESTARTING`	Managed resource cannot fail over to a remote node.
`UNKNOWN`	Status is unrecognized.

FAN Event Format

```
<Event_Type>
VERSION=<n.n>
[service=<serviceName.dbDomainName>]
[database=<dbName>] [instance=<sid>]
[host=<hostname>]
status=<Event_Status>
reason=<Event_Reason>
[card=<n>]
timestamp=<eventDate> <eventTime>
```

```
SERVICE VERSION=1.0 service=ERP.example.com
database=ORCL status=up reason=user card=4
timestamp=16-Jul-2013 13:21:11
```

```
NODE VERSION=1.0 host=host01
status=nodedown timestamp=16-Jul-2013 11:42:05
```

Load Balancing Advisory: FAN Event

Parameter	Description
VERSION	Version of the event payload
EVENT_TYPE	SERVICEMETRICS
SERVICE	Matches DBA_SERVICES
DATABASE	Unique DB name supporting the service
TIMESTAMP	Date and time stamp (local time zone)
INSTANCE	Instance name supporting the service
PERCENT	Percentage of work to send to this database and instance
FLAG	GOOD, VIOLATING, NO DATA, BLOCKED

Server-Side Callouts Implementation

- The callout directory:
  - `<Grid Home>/racg/usrco`
  - Can store more than one callout
  - Grants execution on callout directory and callouts only to the Oracle Clusterware user
- Callout execution order is nondeterministic.
- Writing callouts involves:
  1. Parsing callout arguments: The event payload
  2. Filtering incoming FAN events
  3. Executing event-handling programs

Server-Side Callout Parse: Example

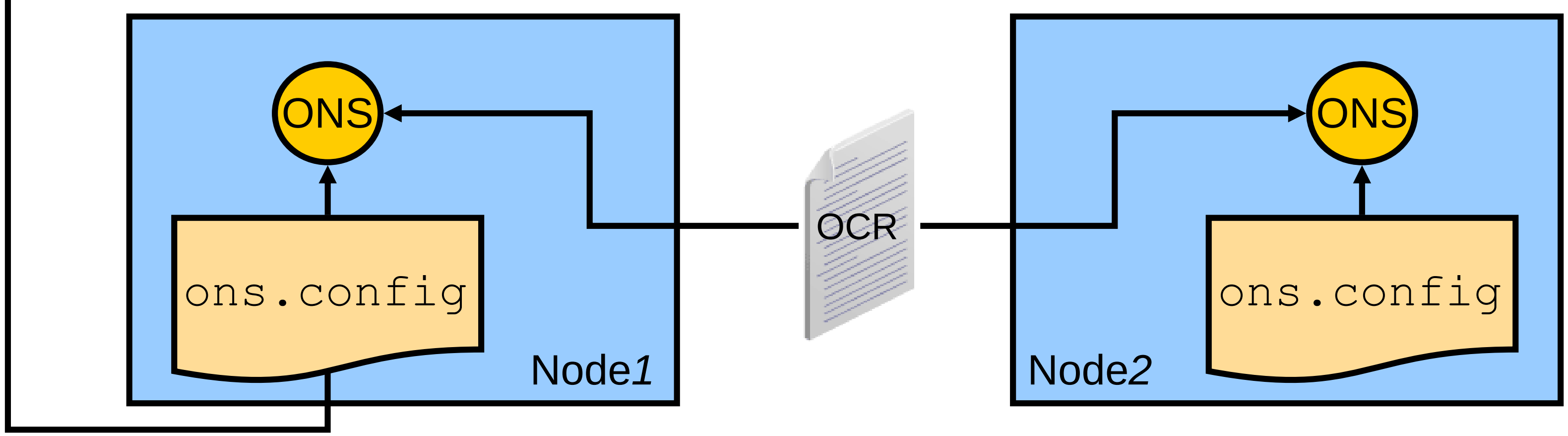
```
#!/bin/sh
NOTIFY_EVENTTYPE=$1
for ARGS in $*; do
   PROPERTY=`echo $ARGS | $AWK -F"=" '{print $1}'`
   VALUE=`echo $ARGS | $AWK -F"=" '{print $2}'`
   case $PROPERTY in
     VERSION|version)     NOTIFY_VERSION=$VALUE ;;
     SERVICE|service)     NOTIFY_SERVICE=$VALUE ;;
     DATABASE|database)   NOTIFY_DATABASE=$VALUE ;;
     INSTANCE|instance)   NOTIFY_INSTANCE=$VALUE ;;
     HOST|host)           NOTIFY_HOST=$VALUE ;;
     STATUS|status)       NOTIFY_STATUS=$VALUE ;;
     REASON|reason)       NOTIFY_REASON=$VALUE ;;
     CARD|card)           NOTIFY_CARDINALITY=$VALUE ;;
     TIMESTAMP|timestamp) NOTIFY_LOGDATE=$VALUE ;;
     ??:??:??)            NOTIFY_LOGTIME=$PROPERTY ;;
   esac
done
```

Server-Side Callout Filter: Example

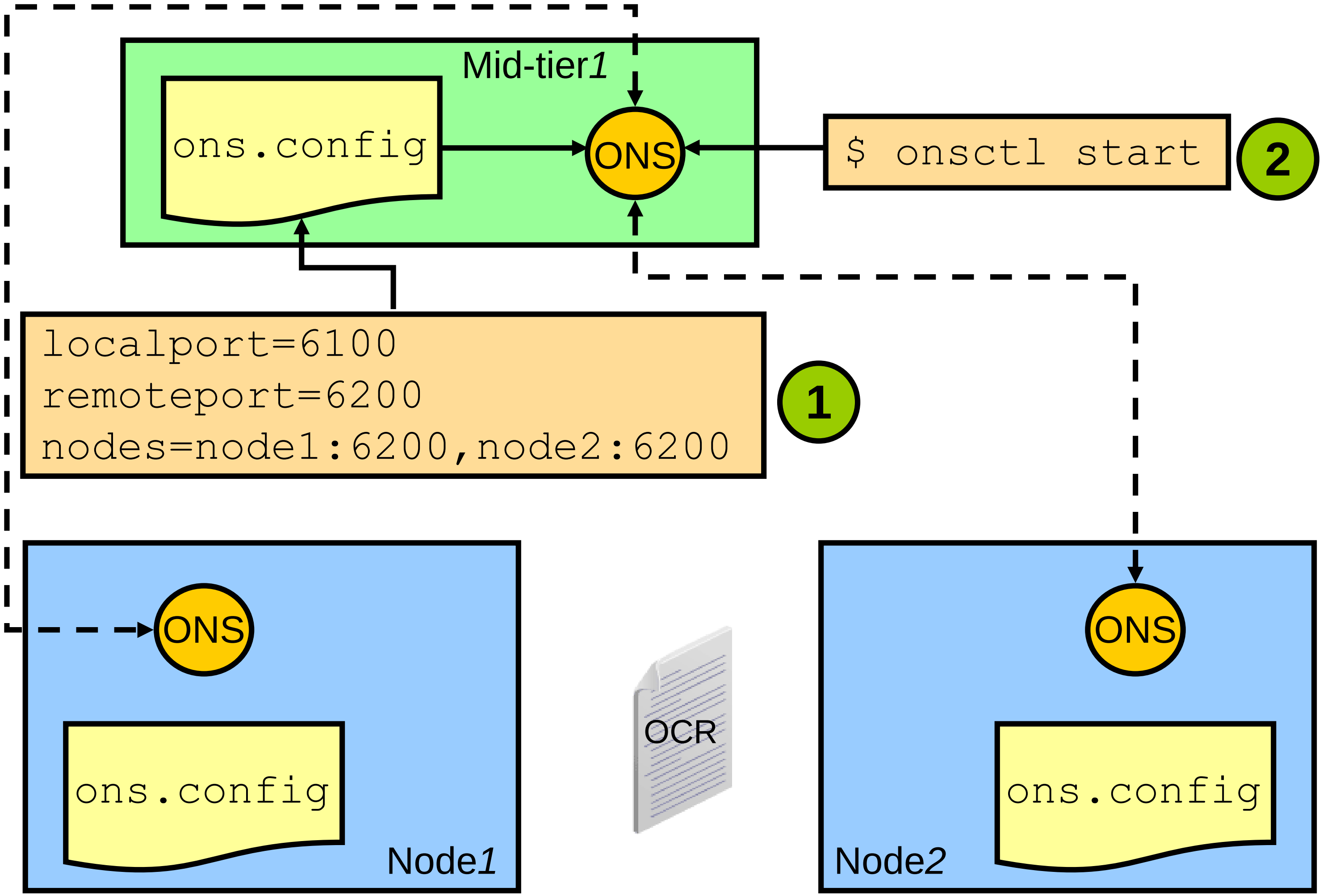
```
if ((( [ $NOTIFY_EVENTTYPE = "SERVICE"          ] ||
       [ $NOTIFY_EVENTTYPE = "DATABASE"         ] ||    \
       [ $NOTIFY_EVENTTYPE = "NODE"             ]       \
     ) &&                                               \
     ( [ $NOTIFY_STATUS = "not_restarting"  ] ||        \
       [ $NOTIFY_STATUS = "restart_failed"  ]           \
     )) &&                                              \
     ( [ $NOTIFY_DATABASE = "HQPROD"            ] ||    \
       [ $NOTIFY_SERVICE  = "ERP"               ]       \
     ))
then
     /usr/local/bin/logTicket $NOTIFY_LOGDATE  \
                              $NOTIFY_LOGTIME  \
                              $NOTIFY_SERVICE  \
                              $NOTIFY_DBNAME   \
                              $NOTIFY_HOST
fi
```

Server-Side ONS

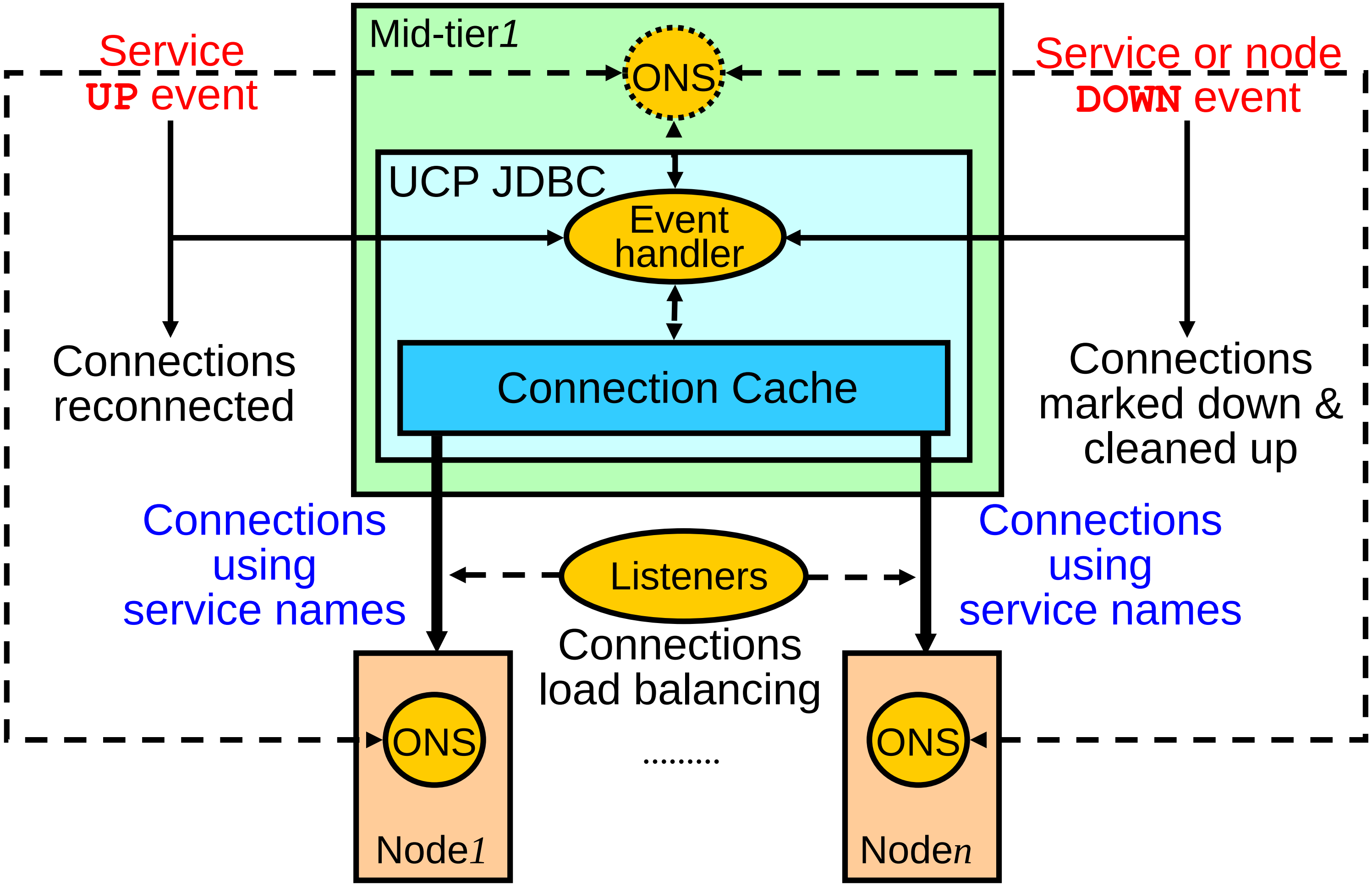
```
usesharedinstall=true
allowgroup=true
localport=6100            # line added by Agent
remoteport=6200           # line added by Agent
nodes=host01:6200         # line added by Agent
```



Optionally Configuring the Client-Side ONS



UCP JDBC Fast Connection Failover: Overview



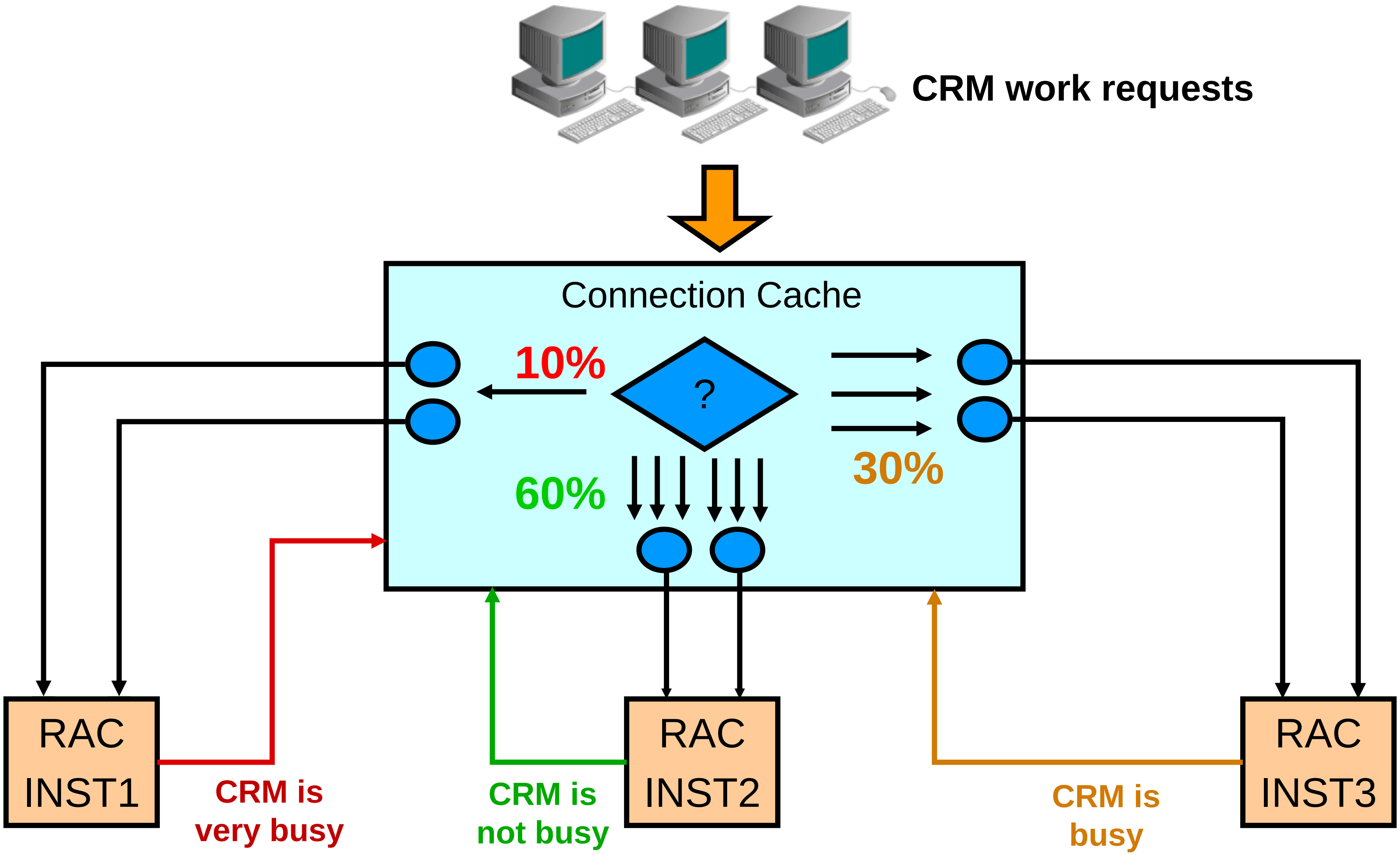
JDBC/ODP.NET FCF Benefits

- Database connections are balanced across preferred instances according to LBA.
- Database work requests are balanced across preferred instances according to LBA.
- Database connections are anticipated.
- Database connection failures are immediately detected and stopped.

Load Balancing Advisory

- The Load Balancing Advisory (LBA) is an advisory for sending work across RAC instances.
- The LBA advice is available to all applications that send work:
  – JDBC and ODP connection pools
  – Connection load balancing
- The LBA advice sends work to where services are executing well and resources are available:
  – Relies on service goodness
  – Adjusts distribution for different power nodes, different priority and shape workloads, and changing demand
  – Stops sending work to slow, hung, or failed nodes

UCP JDBC/ODP.NET Runtime Connection Load Balancing: Overview



Connection Load Balancing in RAC

- Connection load balancing allows scan listeners to distribute connection requests to the best instances.
- This is dependant on the `-clbgoal` setting for the service.
- For connection load balancing, you can use a goal of:
  - `LONG`: Use this connection load balancing method if run-time load balancing is not required. (Typical for batch operations)

```
$ srvctl modify service -db orcl -service batchconn
-clbgoal LONG
```

  - `SHORT`: Use this method for applications that use run-time load balancing or when using connection pools that are integrated with LBA.

```
$ srvctl modify service -db orcl -service oltpapp
-clbgoal SHORT
```

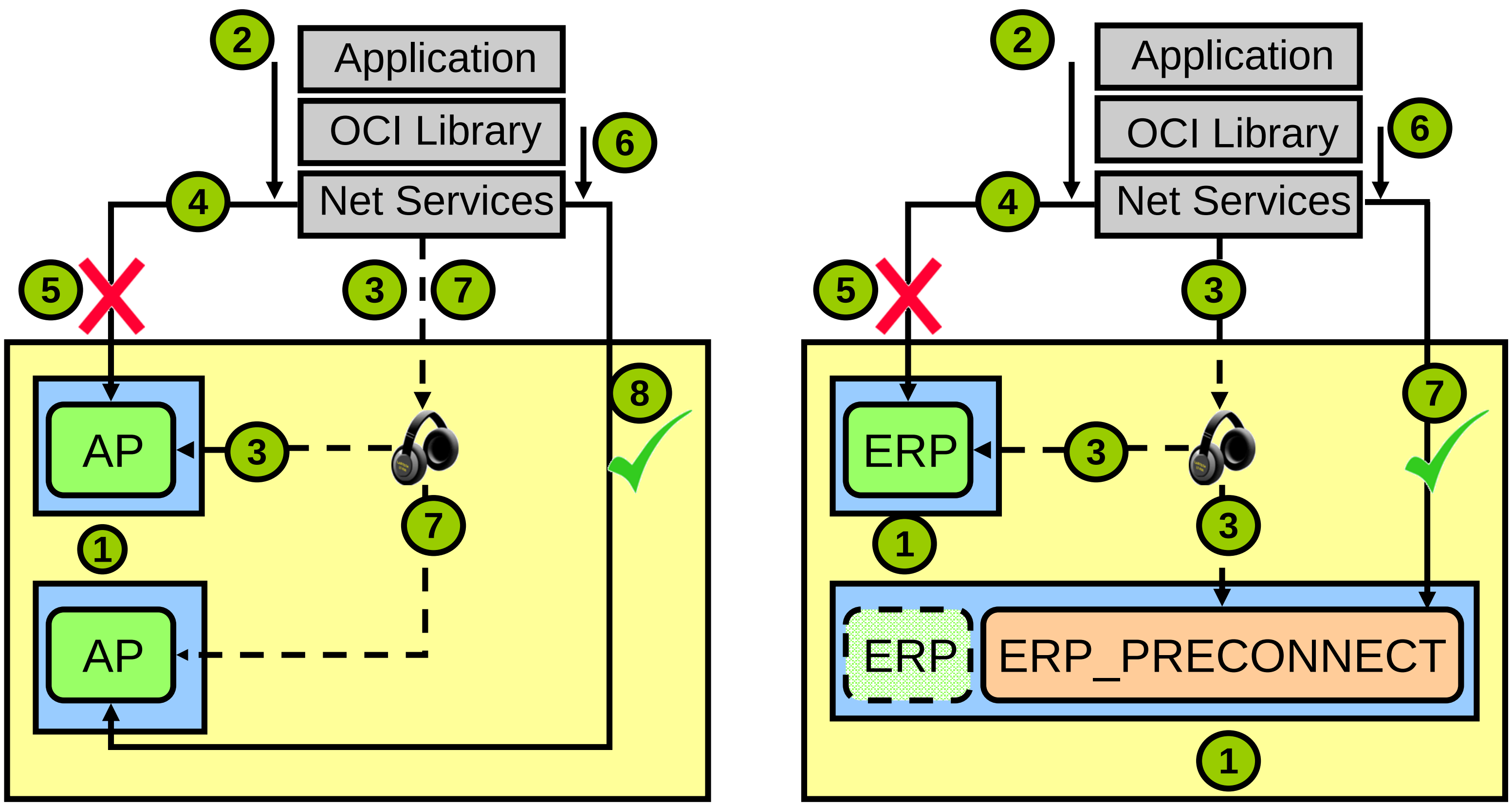
Monitoring LBA FAN Events

```
SQL> SELECT TO_CHAR(enq_time, 'HH:MI:SS') Enq_time, user_data
  2  FROM sys.sys$service_metrics_tab
  3  ORDER BY 1 ;

ENQ_TIME USER_DATA
-------- -----------------------------------------------------
...
04:19:46 SYS$RLBTYP('MYSERV', 'VERSION=1.0 database=orcl
         service=MYSERV { {instance=orcl_2 percent=50
         flag=UNKNOWN}{instance=orcl_1 percent=50 flag=UNKNOWN}
         } timestamp=2013-07-19 11:07:32')
04:20:16 SYS$RLBTYP('MYSERV', 'VERSION=1.0 database=orcl
         service=MYSERV { {instance=orcl_2 percent=80
         flag=UNKNOWN}{instance=orcl_1 percent=20 flag=UNKNOWN}
         } timestamp=2013-07-19 11:08:11')
SQL>
```

Transparent Application Failover: Overview

TAF Basic

1 TAF Preconnect



TAF Basic Configuration on Server-Side: Example

```
$ srvctl add service -db RACDB -service APSVC
   -failovermethod BASIC -failovertype SELECT
   -failoverretry 10 -failoverdelay 30 -serverpool sp1
$ srvctl start service -db RACDB -service APSVC
```

```
apsvc =
 (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = cluster01-scan)
    (PORT = 1521))
    (CONNECT_DATA =
      (SERVICE_NAME = apsvc)))
```

```
$ sqlplus AP/AP@apsvc
SQL> SELECT inst_id, username, service_name, failover_type,
            failover_method
     FROM gv$session WHERE username='AP';

INST_ID USERNAME SERVICE_NAME FAILOVER_TYPE FAILOVER_M
------- -------- ------------ ------------- ----------
      1       AP        apsvc        SELECT      BASIC
```

TAF Basic Configuration on a Client-Side: Example

```
$ srvctl add service -db RACDB -service AP -serverpool sp1
```

```
$ srvctl start service -db RACDB -service AP
```

```
AP =
 (DESCRIPTION =(FAILOVER=ON)(LOAD_BALANCE=ON)
  (ADDRESS=(PROTOCOL=TCP)(HOST=N1VIP)(PORT=1521))
  (ADDRESS=(PROTOCOL=TCP)(HOST=N2VIP)(PORT=1521))
   (CONNECT_DATA =
      (SERVICE_NAME = AP)
      (FAILOVER_MODE= (TYPE=select)
                      (METHOD=basic)
                      (RETRIES=20)
                      (DELAY=15))))
```

TAF Preconnect Configuration: Example

```
$ srvctl add service -db RACDB -service ERP
       -preferred I1 -available I2
       -tafpolicy PRECONNECT

$ srvctl start service -db RACDB -service ERP
```

```
ERP =
 (DESCRIPTION =(FAILOVER=ON)(LOAD_BALANCE=ON)
  (ADDRESS=(PROTOCOL=TCP)(HOST=N1VIP)(PORT=1521))
  (ADDRESS=(PROTOCOL=TCP)(HOST=N2VIP)(PORT=1521))
   (CONNECT_DATA = (SERVICE_NAME = ERP)
    (FAILOVER_MODE = (BACKUP=ERP_PRECONNECT)
                (TYPE=SESSION)(METHOD=PRECONNECT))))

ERP_PRECONNECT =
 (DESCRIPTION =(FAILOVER=ON)(LOAD_BALANCE=ON)
  (ADDRESS=(PROTOCOL=TCP)(HOST=N1VIP)(PORT=1521))
  (ADDRESS=(PROTOCOL=TCP)(HOST=N2VIP)(PORT=1521))
   (CONNECT_DATA = (SERVICE_NAME = ERP_PRECONNECT)))
```

TAF Verification

```
SELECT   machine, failover_method, failover_type,
         failed_over, service_name, COUNT(*)
FROM     v$session
GROUP BY machine, failover_method, failover_type,
         failed_over, service_name;
```

1st node

```
MACHINE FAILOVER_M FAILOVER_T FAI SERVICE_N COUNT(*)
------- ---------- ---------- --- --------  --------
node1   BASIC      SESSION    NO  AP        1
node1   PRECONNECT SESSION    NO  ERP       1
```

2nd node

```
MACHINE FAILOVER_M FAILOVER_T FAI SERVICE_N COUNT(*)
------- ---------- ---------- --- --------- --------
node2   NONE       NONE       NO  ERP_PRECO 1
```

2nd node after

```
MACHINE FAILOVER_M FAILOVER_T FAI SERVICE_N COUNT(*)
------- ---------- ---------- --- --------  --------
node2   BASIC      SESSION    YES AP        1
node2   PRECONNECT SESSION    YES ERP_PRECO 1
```

FAN Connection Pools and TAF Considerations

- Both techniques are integrated with services and provide service connection load balancing.
- Connection pools that use FAN are always preconnected.
- TAF may rely on operating system (OS) timeouts to detect failures.
- FAN never relies on OS timeouts to detect failures.

Introducing Transaction Guard and Application Continuity

Oracle Database 12*c* introduces two fundamental capabilities for ensuring continuity of applications after database outages :

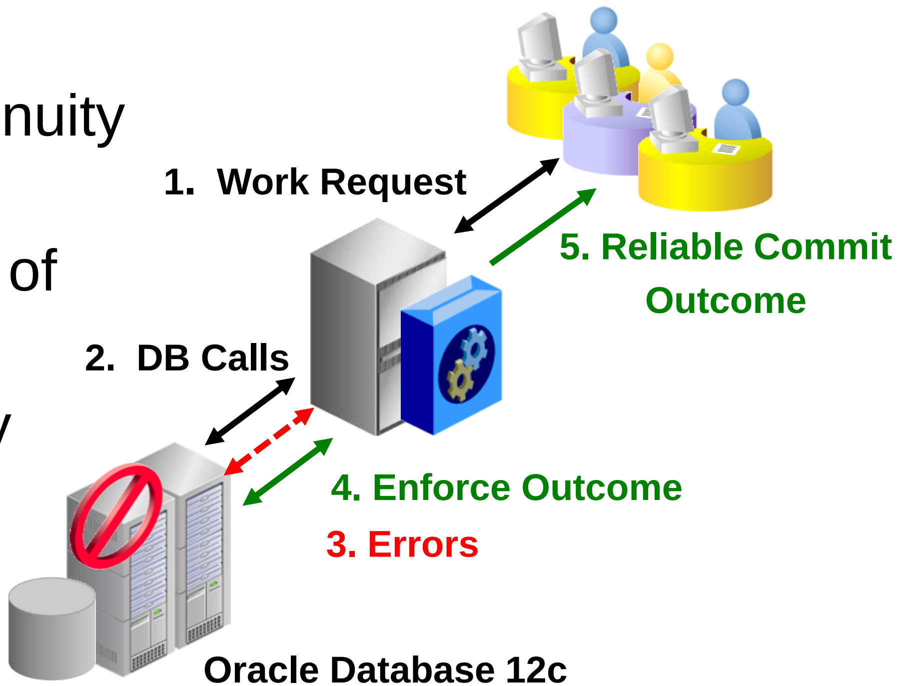
- Transaction Guard: A new reliable protocol and API that returns the outcome of the last transaction after a recoverable error has occurred
- Application Continuity: A feature that attempts to mask database session outages by recovering the in-flight work for requests submitted to the database

What Is Transaction Guard?

Transaction Guard:

- Is a tool that provides a reliable commit outcome for the last transaction after errors
- Is an API available for JDBC Thin, C/C++ (OCI/OCCI), and ODP.NET
- Is used by Application Continuity for at-most-once execution
- Can be used independently of Application Continuity

Without Transaction Guard, retry can cause logical corruption.

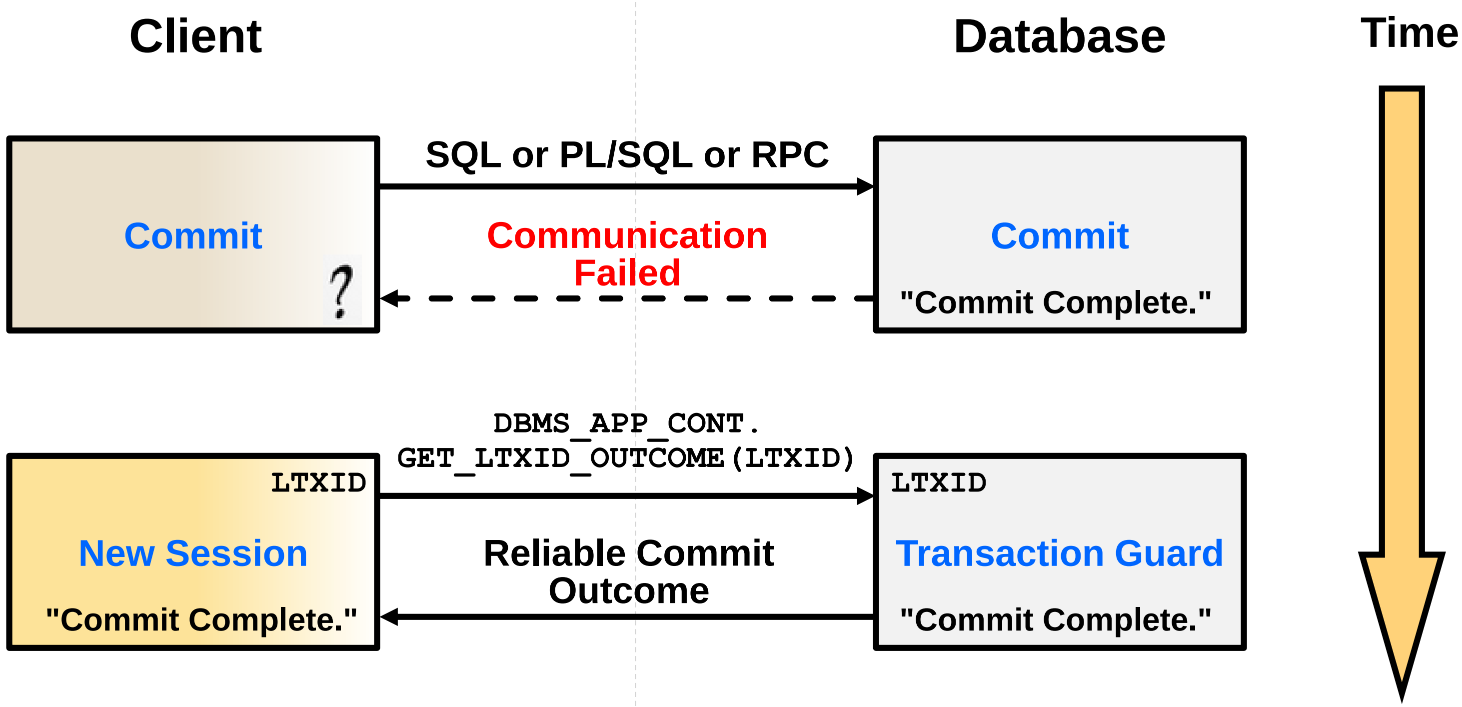


Benefits of Transaction Guard

- After outages, users know what happened to their in-flight transactions, such as fund transfers, flight bookings, and bill payments.
- Transaction Guard provides better performance and reliability than home-built code for idempotence.

How Transaction Guard Works

Transaction Guard is a reliable protocol and API that enables applications to know the outcome of the last transaction.



Using Transaction Guard

- Supported transaction types:
  - Local commit
  - Auto-commit and Commit on Success
  - Commit embedded in PL/SQL
  - DDL, DCL, and Parallel DDL
  - Remote, Distributed commit
- Not supported in release 12.1:
  - XA
  - Read-write database links from Active Data Guard
- Server configuration:
  - Set the `COMMIT_OUTCOME=TRUE` service attribute
  - Optionally, set the `RETENTION_TIMEOUT` service attribute
- Supported clients:
  - JDBC Thin, OCI, OCCI, and ODP.NET

Creating Services for Transaction Guard

- To create a service using Transaction Guard but not Application continuity:

```
$ srvctl add service -db racdb -service app3
   -serverpool Srvpool1
   -commit_outcome TRUE          Mandatory Settings for Transaction Guard
   -retention 86400 -failoverretry 30 -failoverdelay 10
   -notification TRUE -rlbgoal SERVICE_TIME -clbgoal SHORT
```

- To modify an existing service to enable Transaction Guard:

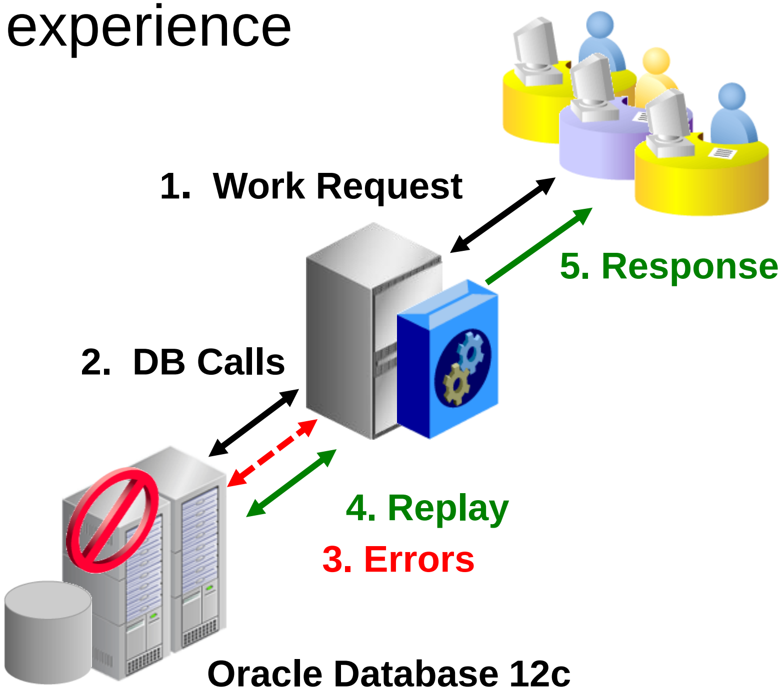
```
$ srvctl modify service -db racdb -service app4
   -commit_outcome TRUE -retention 86400
   -notification TRUE
```

- To use Transaction Guard, a DBA must grant permission:

```
SQL> GRANT execute ON DBMS_APP_CONT;
```

What Is Application Continuity?

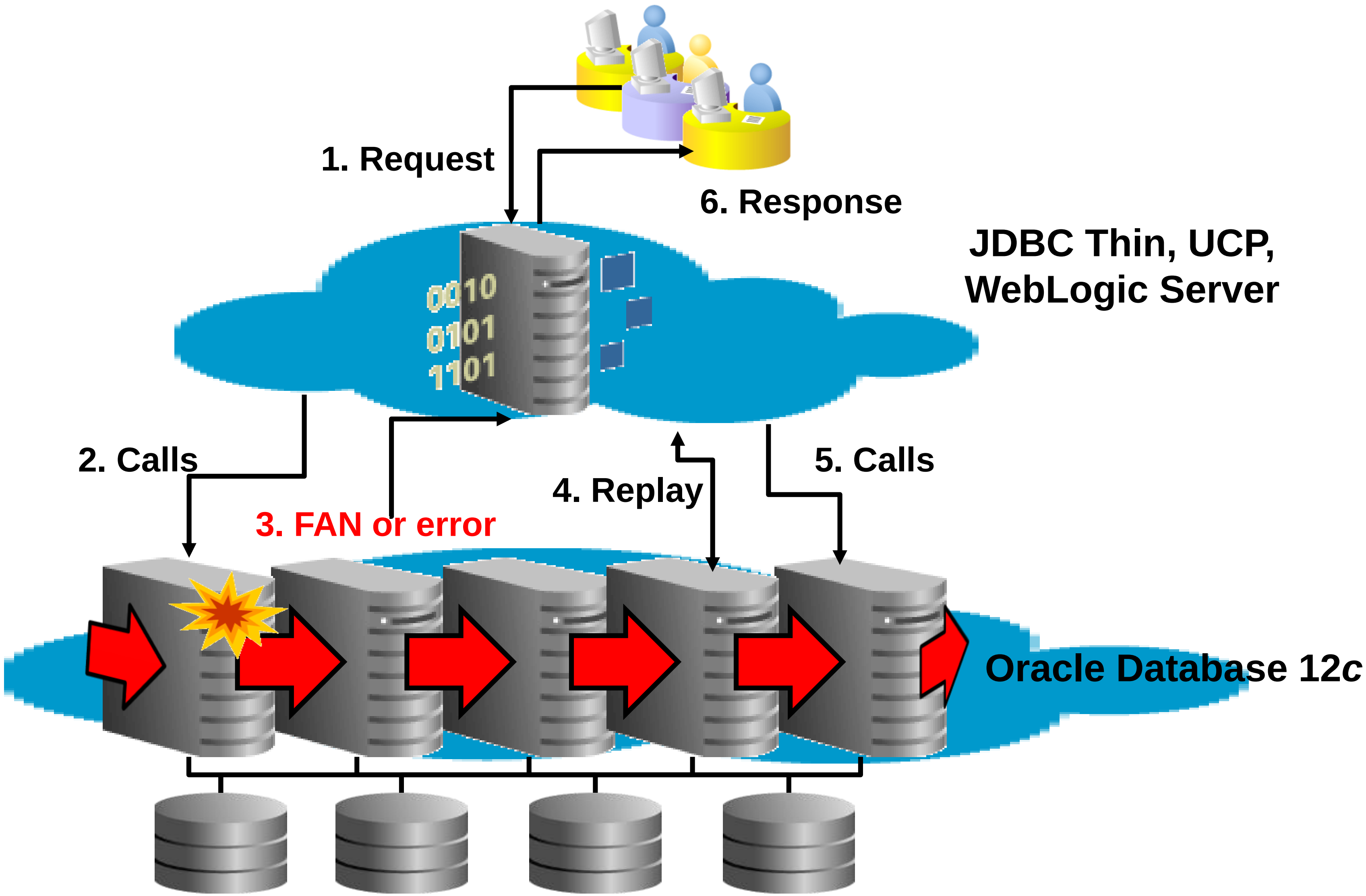
- Replays in-flight work on recoverable errors
- Masks many hardware, software, network, storage errors, and outages, when successful
- Improves the end-user experience



Benefits of Application Continuity

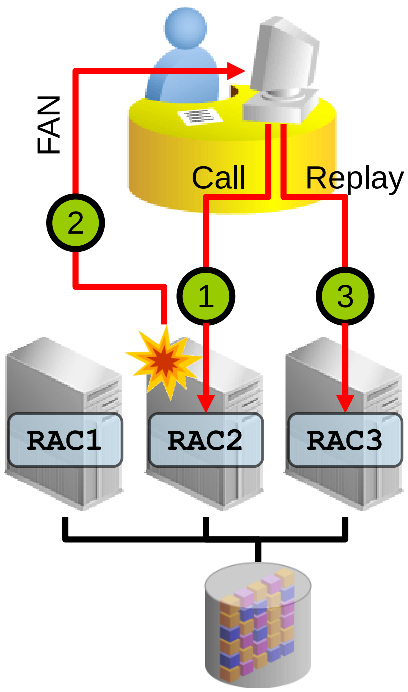
- Uninterrupted user service, when replay is successful
- Can help relocate database sessions to remaining servers for planned outages
- Improves developer productivity by masking outages that can be masked
- Few or no application changes
- Simple configuration

How Does Application Continuity Work?



RAC and Application Continuity

- Application Continuity transparently replays database requests after a failed session.
  – Users are shielded from many types of problems.
- Using Application Continuity with RAC provides:
  – Protection against a wider variety of failure scenarios
  – Faster reconnect and replay
  – Request replay on another RAC instance



Using Application Continuity

- Supported database operations:
  - SQL, PL/SQL, and JDBC RPC: `SELECT`, `ALTER SESSION`, DML, DDL, `COMMIT`, `ROLLBACK`, `SAVEPOINT`, and JDBC RPCs
  - Transaction models: Local, Parallel, Remote, Distributed, and Embedded PL/SQL
  - Mutable functions
  - Transaction Guard
- Works in conjunction with:
  - Oracle RAC and RAC One
  - Oracle Active Data Guard
- Hardware acceleration on current Intel and SPARC chips
- Supported clients:
  - JDBC Thin driver, Universal Connection Pool, and WebLogic Server

Creating Services for Application Continuity

- To create a service for Application Continuity for a policy-managed RAC database:

```
$ srvctl add service -db racdb -service app2
  -serverpool Srvpool1
  -failovertype TRANSACTION
  -commit_outcome TRUE
  -replay_init_time 1800 -failoverretry 30 -failoverdelay 10
  -retention 86400
  -notification TRUE -rlbgoal SERVICE_TIME -clbgoal SHORT
```

Mandatory Settings for Application Continuity

Optional Settings for Application Continuity

- To modify an existing service for Application Continuity:

```
$ srvctl modify service -db racdb -service app1 -clbgoal SHORT
  -rlbgoal SERVICE_TIME -failoverretry 30 -failoverdelay 10
  -failovertype TRANSACTION -commit_outcome TRUE
  -replay_init_time 1800 -retention 86400 -notification TRUE
```

Quiz

Which of the following are benefits of implementing Fast Application Notification?

a. No need for connections to rely on connection timeouts
b. Used by Load Balancing Advisory to efficiently propagate load information
c. Database connection failures are immediately detected and stopped.
d. Designed for enterprise application and management console integration

Quiz

Application Continuity attempts to mask recoverable database outages from applications and users by restoring database sessions and replaying database calls.

a. True

b. False

Summary

In this lesson, you should have learned how to:

- Configure client-side connect-time load balancing
- Configure client-side connect-time failover
- Configure server-side connect-time load balancing
- Configure Transparent Application Failover (TAF)
- Describe the purpose of Transaction Guard and Application Continuity
- Describe the key concepts relating to Application Continuity

Practice 10 Overview: Using Application Continuity

This practice covers using Application Continuity against a RAC database to demonstrate how Application Continuity helps an application to seamlessly recover after the failure of a RAC instance.

Oracle RAC One Node

Objectives

After completing this lesson, you should be able to:

- Perform an online database relocation
- Add an Oracle RAC One Node Database to an existing Cluster
- Convert an Oracle RAC One Node database to a RAC database
- Use DBCA to convert a single instance database to a RAC One Node database

Oracle RAC One Node

- Oracle RAC One Node is a single instance of a RAC-enabled database running on one node in the cluster only.
- This adds to the flexibility for database consolidation while reducing management overhead by providing a standard deployment for Oracle databases in the enterprise.
- Oracle RAC One Node requires Grid Infrastructure and, requires the same hardware setup as a RAC database.
- If applications require more resources than a single node can supply, you can upgrade online to Oracle RAC.
- If the node running Oracle RAC One Node becomes overloaded, you can relocate the instance to another node.

Creating an Oracle RAC One Node Database

- You can create Oracle RAC One Node databases by using DBCA.
- Single instance or RAC databases can be converted to Oracle RAC One Node.
- For Oracle RAC One Node databases, you must configure at least one dynamic database service.
- With an administrator-managed RAC One Node database, service registration is performed as with any other RAC database.

Verifying an Existing RAC One Node Database

```
[oracle@host01 ~]$ srvctl config database -db orcl
Database unique name: orcl
Database name: orcl
Oracle home: /u01/app/oracle/product/12.1.0/dbhome_1
Oracle user: oracle
Spfile: +DATA/orcl/spfileorcl.ora
Password file: +DATA/orcl/orapworcl
Domain: cluster01.example.com
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: orcldb
Database instances:
Disk Groups: DATA
Mount point paths:
Services: SVC1
Type: RACOneNode
Online relocation timeout: 30
Instance name prefix: orcl
Candidate servers: host01,host02
...
```

Information specific to RAC One Node

Oracle RAC One Node Online Relocation

- Oracle RAC One Node allows the online relocation of the database from one server to another.
- The relocation period can be customized up to 12 hours.
- Use the `srvctl relocate database` command to initiate relocation of an Oracle RAC One Node database:

```
srvctl relocate database -db db_unique_name {[-node target]
[-timeout timeout_value] | -abort [-revert]} [-verbose]

-db <db_unique_name> Unique name of database to relocate
-node <target> Target node to which to relocate database
-timeout <timeout> Online relocation timeout in minutes
-abort Abort failed online relocation
-revert Remove target node of failed online relocation request
from the candidate server list of administrator-managed RAC One
Node database
-verbose Verbose output
-help Print usage
```

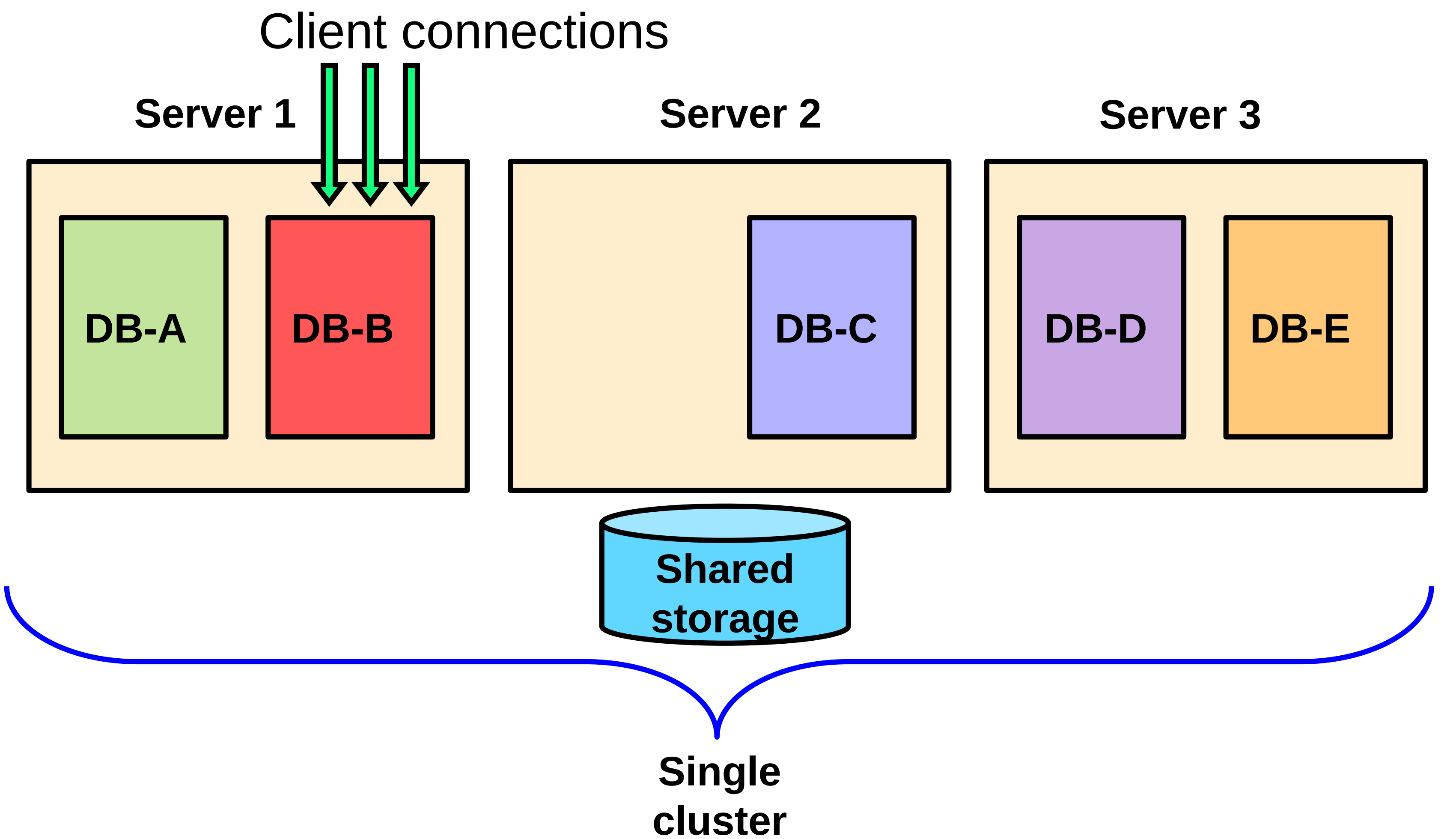
Online Relocation Considerations

- Use either Application Continuity and Oracle Fast Application Notification or Transparent Application Failover to minimize the impact of a relocation on the client.
- If FAN or TAF is not used, any in-flight transactions will be allowed to complete within the timeout value constraint.
  - If the timeout is exceeded, clients will receive an ORA-3113 error when the session is terminated due to the shutdown of the instance.
- If the shutdown of the original instance takes longer than the timeout value, the instance is aborted.
  - The new instance will then perform recovery to clean up any transactions that were aborted due to the shutdown.

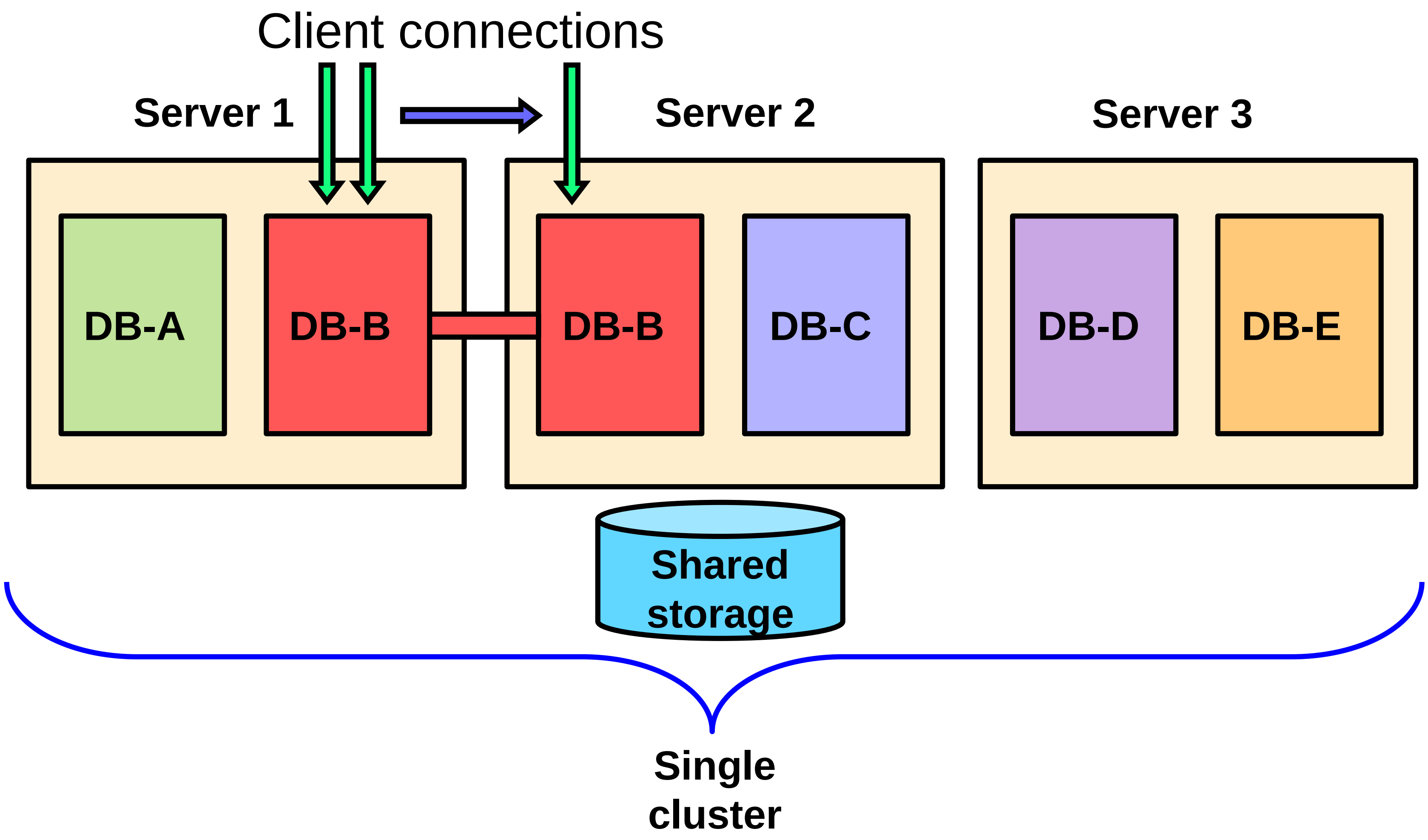
Performing an Online Relocation

```
[oracle@host01 ~]$ srvctl relocate database -db orcl \
 -node host02 -timeout 15 -verbose

Configuration updated to two instances
Instance orcl_2 started
Services relocated
Waiting for 15 minutes for instance orcl_1 to stop.....
Instance orcl_1 stopped
Configuration updated to one instance
```

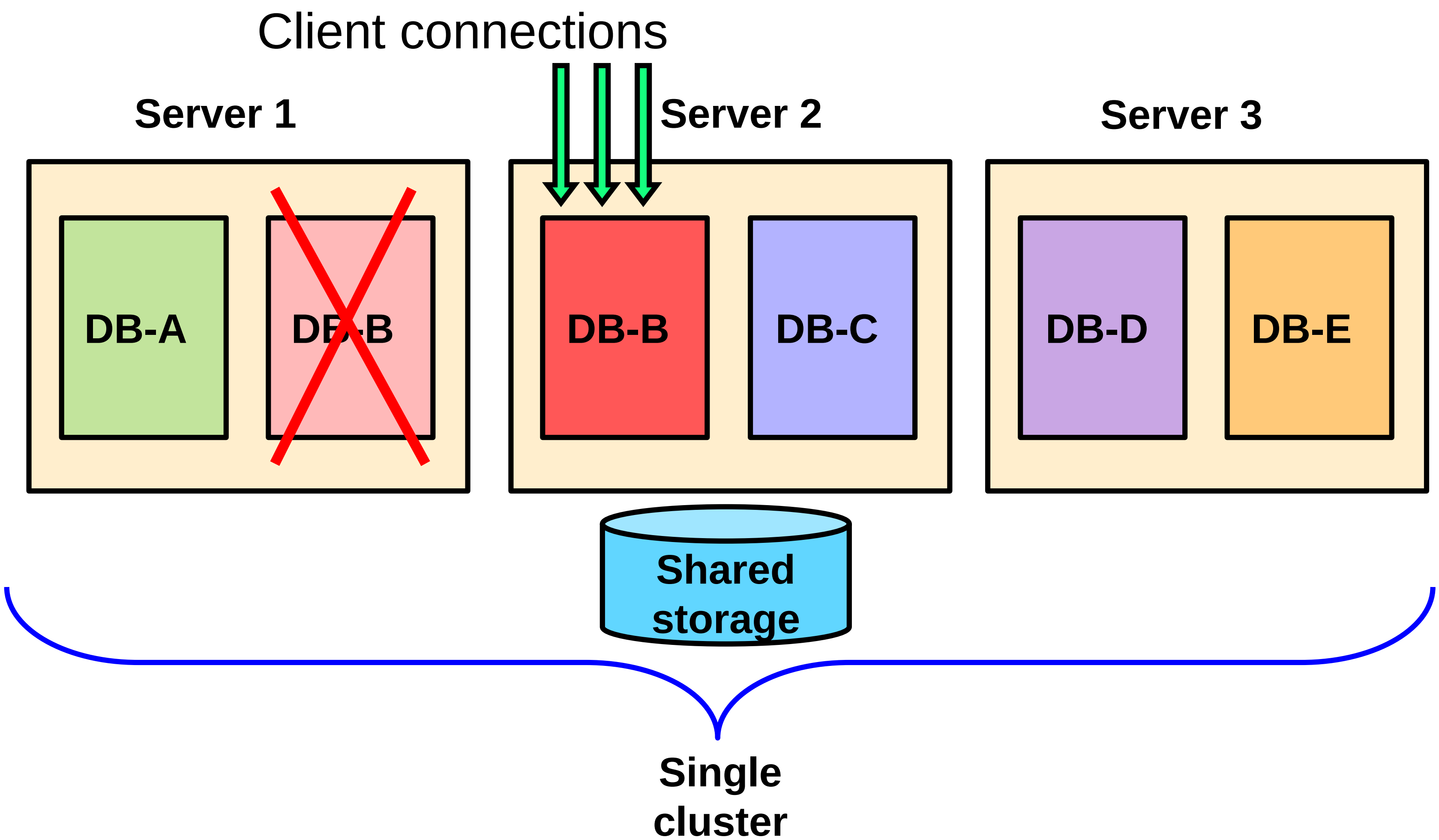
Online Relocation Illustration



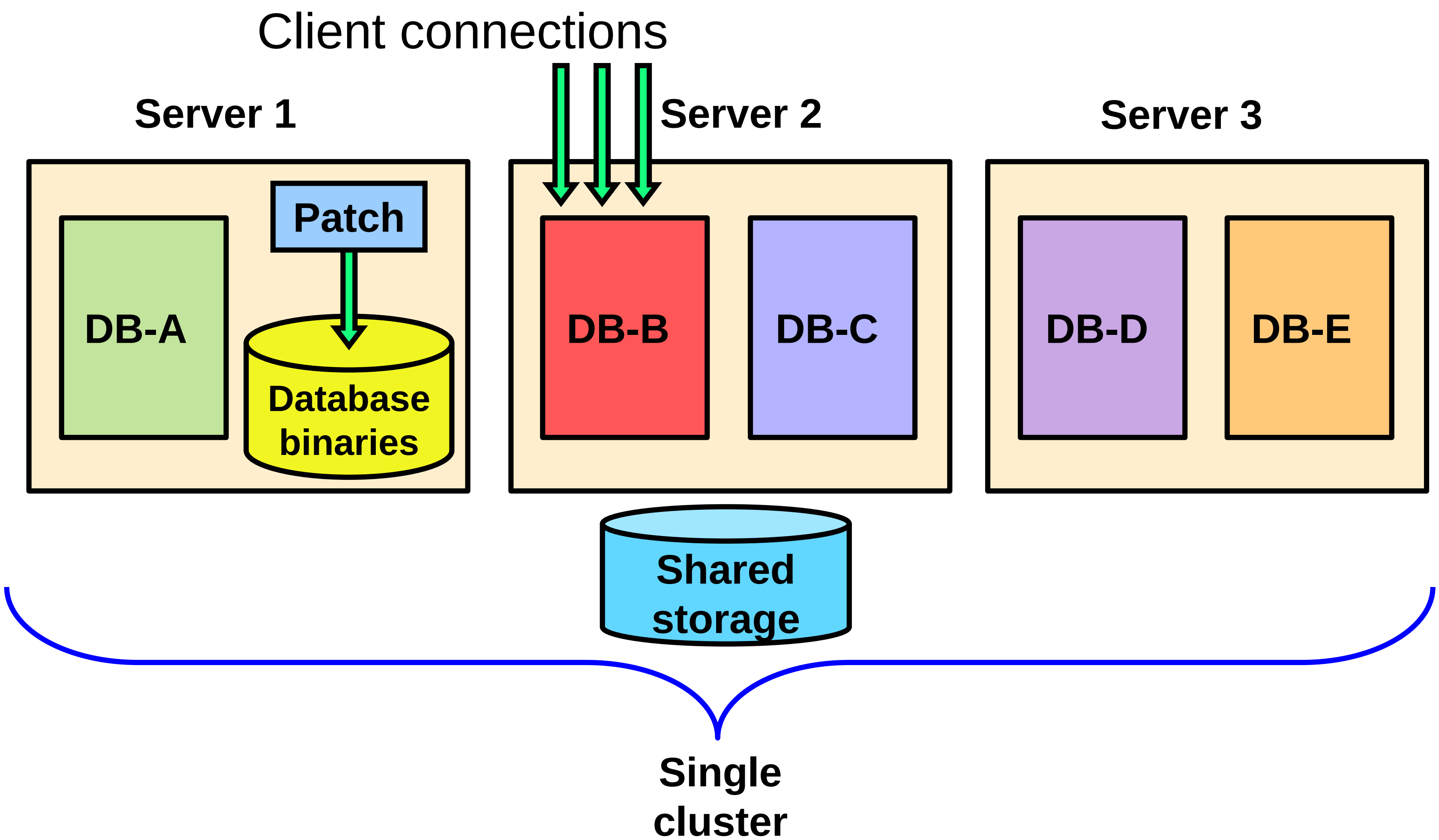
Online Relocation Illustration



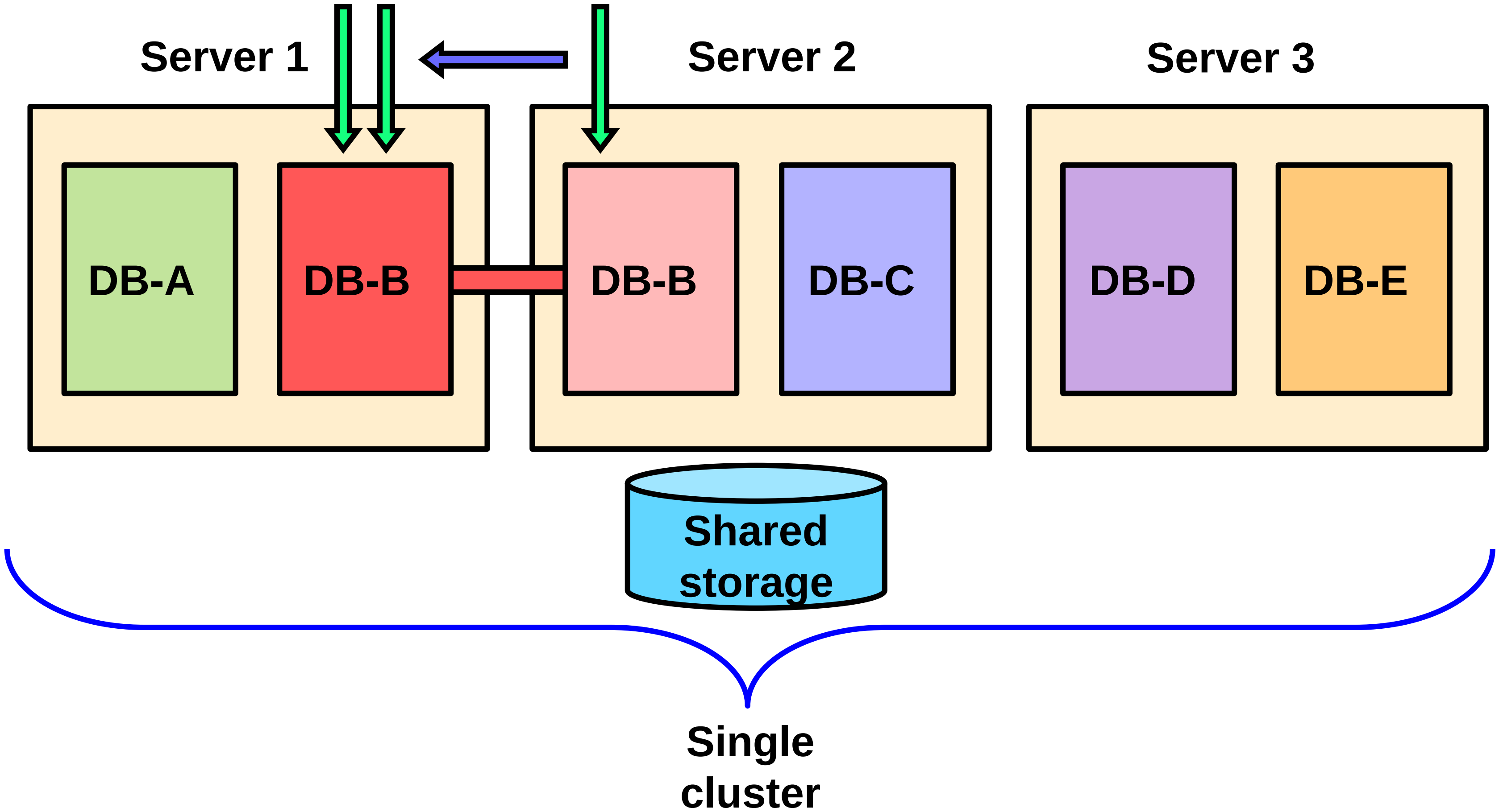
Online Relocation Illustration



Online Maintenance: Rolling Patches



Online Maintenance: Rolling Patches



Adding an Oracle RAC One Node Database to an Existing Cluster

- Use the `srvctl add database` command to add an Oracle RAC One Node database to an existing cluster.

```
srvctl add database -dbtype RACONENODE [-server server_list]
[-instance instance_name] [-timeout timeout_value]
```

- When adding an administrator-managed Oracle RAC One Node database, you can optionally supply an instance prefix with the `-instance instance_name` option of the `srvctl add database` command.
- Each service is configured by using the same value for the `SERVER_POOLS` attribute as the underlying database.
  - When you add services to an Oracle RAC One Node database, `srvctl` configures those services using the value of the `SERVER_POOLS` attribute.

Converting a RAC One Node Database to RAC

To convert a RAC One Node database to RAC:

1. Execute the `srvctl convert database` command.

```
srvctl convert database -db <db_unique_name> -dbtype RAC
[-node <node_1>]
```

2. Create server pools for each service that the database has, in addition to the database server pool.
3. Add the instances on the remaining nodes with the `srvctl add instance` command.

```
srvctl add instance -db <db_unique_name> -instance
instance_name -node <node_2>
srvctl add instance -db <db_unique_name> -instance
instance_name -node <node_n>
```

Converting a RAC One Node Database to RAC

```
$ srvctl convert database -db orcl -dbtype RAC -node host01

$ srvctl add instance -db orcl –instance orcl_2 –node host02

$ srvctl start instance -db orcl -instance orcl_2

$ srvctl config database -d orcl
Database unique name: orcl
Database name: orcl
Oracle home: /u01/app/oracle/product/12.1.0/dbhome_1
Oracle user: oracle
Spfile: +DATA/orcl/spfileorcl.ora
Domain:
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: orcl
Database instances: orcl_1,orcl_2
Disk Groups: DATA,FRA
Services: SERV1
Type: RAC
Database is administrator managed
```

Converting a Single Instance Database to RAC One Node

- Use DBCA to convert from single-instance Oracle databases to Oracle RAC One Node.
- Before you use DBCA to convert a single-instance database to an Oracle RAC One Node database, ensure that your system meets the following conditions:
  - It is a supported hardware and operating system software configuration.
  - The nodes have access to shared storage.
  - Your applications have no design characteristics that preclude their use in a clustered environment.

Converting a RAC Database to RAC One Node

- When converting a RAC database to RAC One Node, first ensure that the RAC database has only one instance.
- If the RAC database is admin-managed, change the configuration of all services to set the preferred instance to the one you want to keep as an RAC One Node database.
  - If a service had a `PRECONNECT` TAF policy, then the policy must be updated to `BASIC` or `NONE` before conversion.
- If the RAC database is policy managed, then change the configuration of all services so they use the same server pool before you convert the RAC database.
- Use the `srvctl convert database` command to convert a RAC database to RAC One Node:

```
srvctl convert database -db db_unique_name -dbtype
RACONENODE [-instance instance_name -timeout timeout]
```

Quiz

The `srvctl add database` command is used to add an Oracle RAC One Node database to an existing cluster.

a. True
b. False

Quiz

Which of the following conditions must be met before a single instance database can be converted to a RAC One Node Database? (Choose three)

a. Your environment is a supported hardware and operating system software configuration.
b. It has shared storage: either Oracle Cluster File System or Oracle ASM is available and accessible from all nodes.
c. You must disable Oracle Clusterware on all nodes.
d. Your applications have no design characteristics that preclude their use in a clustered environment.

Summary

In this lesson, you should have learned how to:

- Perform an online database relocation
- Add an Oracle RAC One Node Database to an existing cluster
- Convert an Oracle RAC One Node database to a RAC database
- Use DBCA to convert a single instance database to a RAC One Node database.

Practice 11: Overview

The practices for this lesson cover the following topics:

- RAC One Node Database creation using DBCA
- RAC One Node online relocation
- Converng a RAC One Node database to a RAC Database

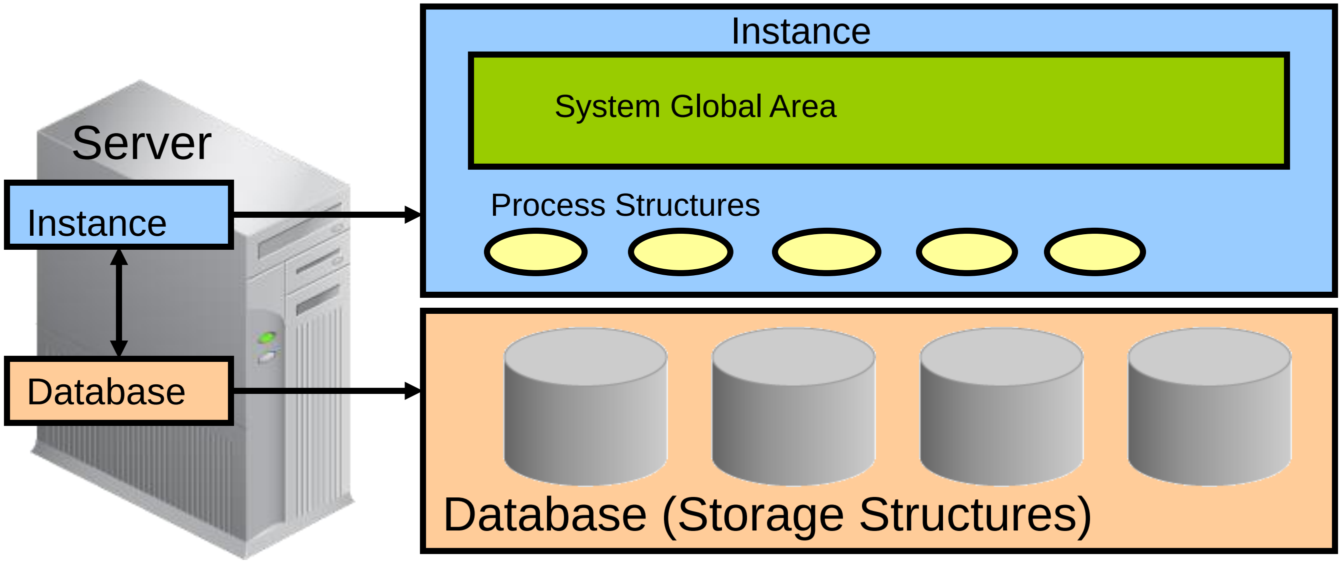
Multitenant Architecture and RAC

Objectives

After completing this lesson, you should be able to do the following:

- Describe the multitenant architecture in a non-RAC environment
- Describe the multitenant architecture in a RAC environment
- Create a RAC multitenant container database (CDB)
- Create a pluggable database (PDB) in a RAC CDB
- Use default CDB and PDB services
- Create PDB services to associate PDB services with server pools
- Drop a pluggable database (PDB) from a RAC CDB

Non-CDB Architecture

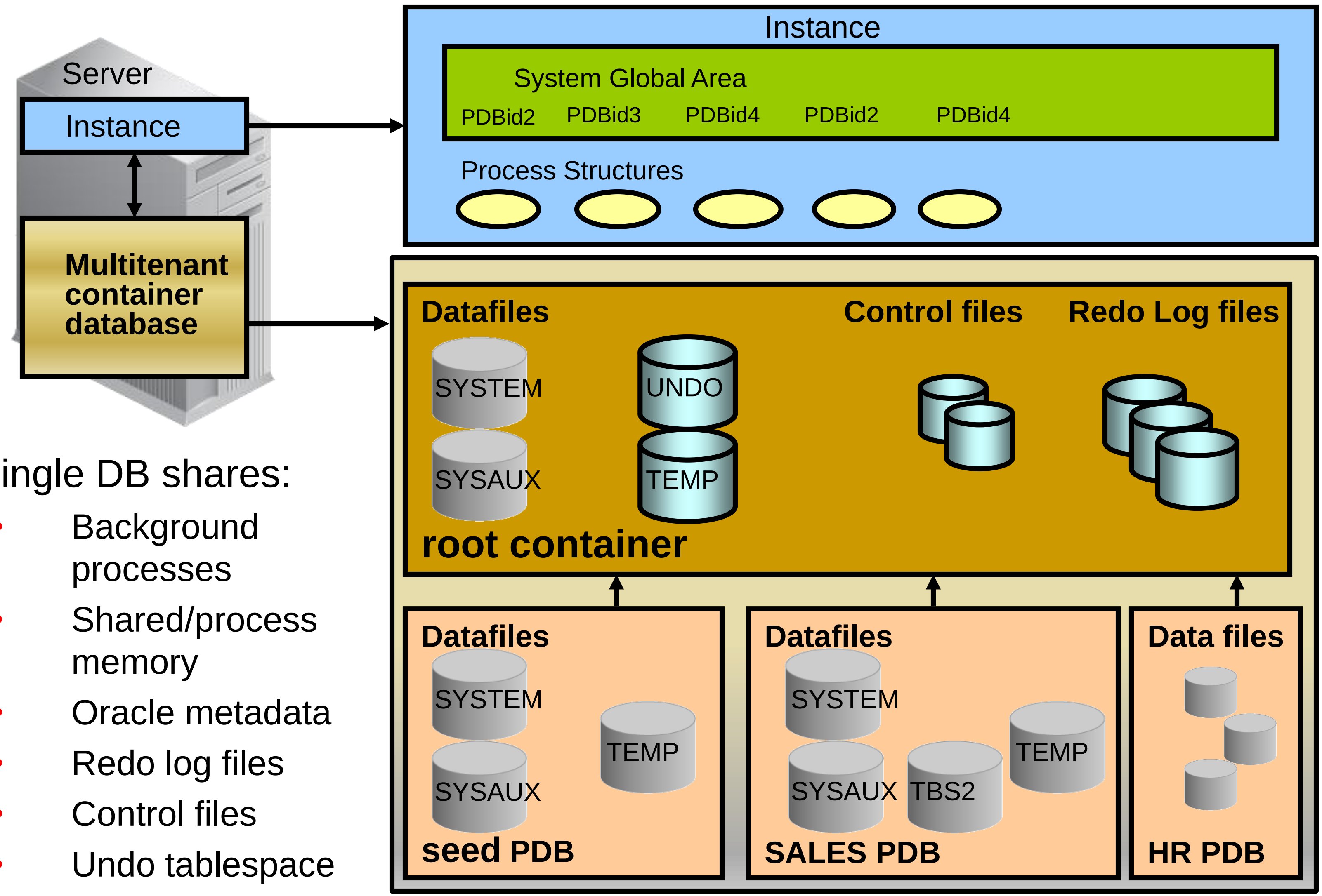


- Multiple non-CDBs share nothing:
  - Too many background processes
  - High shared/process memory
  - Many copies of Oracle metadata

Multitenant Architecture: Benefits

- Operates **multiple databases in a centrally managed platform** to lower costs:
  - Less instance overhead
  - Less storage cost
- Reduces DBA resources costs and maintains security
  - No application changes
  - **Fast and easy provisioning**
  - **Time saving for patching and upgrade**
  - **Maintain separation of duties** between:
    - Different application administrators
    - Application administrators and DBA
    - Users within application
- **Provides isolation**

CDB in a Non-RAC Environment



Single DB shares:

- Background processes
- Shared/process memory
- Oracle metadata
- Redo log files
- Control files
- Undo tablespace

Containers

Two types of containers in `V$CONTAINERS`:

- The root container
  - The first mandatory container created at CDB creation
  - Oracle system–supplied common objects and metadata
  - Oracle system–supplied common users and roles
- Pluggable database containers (PDBs)
  - A container for an application:
    - Tablespaces (permanent and temporary)
    - Schemas/Objects/Privileges
    - Created/cloned/unplugged/plugged
  - Particular seed `PDB$SEED`: fast provisioning of a new PDB
  - Limit of 253 PDBs in a CDB including the seed
  - Limit of 1024 services in a CDB

Terminology

- DBA, CDBA, and PDBA
- Common vs Local:
  - Users
    - A common user can only be created in the root container.
    - A local user can only be created and known in a PDB.
  - Roles
    - A common role can only be created in the root container.

```
SQL> CREATE ROLE C##manager
     CONTAINER=ALL;
```

    - A local role can only be created and known in a PDB.
  - Privileges
- CDB vs PDB level

CDB1

root

View CDB_USERS

USERNAME	COMMON
SYS	YES
SYSTEM	YES
C##JIM	**YES**
DBSNMP	YES
HR_MGR	NO
SALES	NO

```
SQL> CREATE USER C##JIM
IDENTIFIED BY x CONTAINER=ALL;
```

PDB_HR

View DBA_USERS

USERNAME	COMMON
SYS	YES
SYSTEM	YES
C##JIM	**YES**
DBSNMP	YES
HR_MGR	**NO**

Data Dictionary Views

`CDB_xxx` All objects in the multitenant container database across all PDBs

`DBA_xxx` All of the objects in a container or pluggable database

`ALL_xxx` Objects accessible by the current user

`USER_xxx` Objects owned by the current user

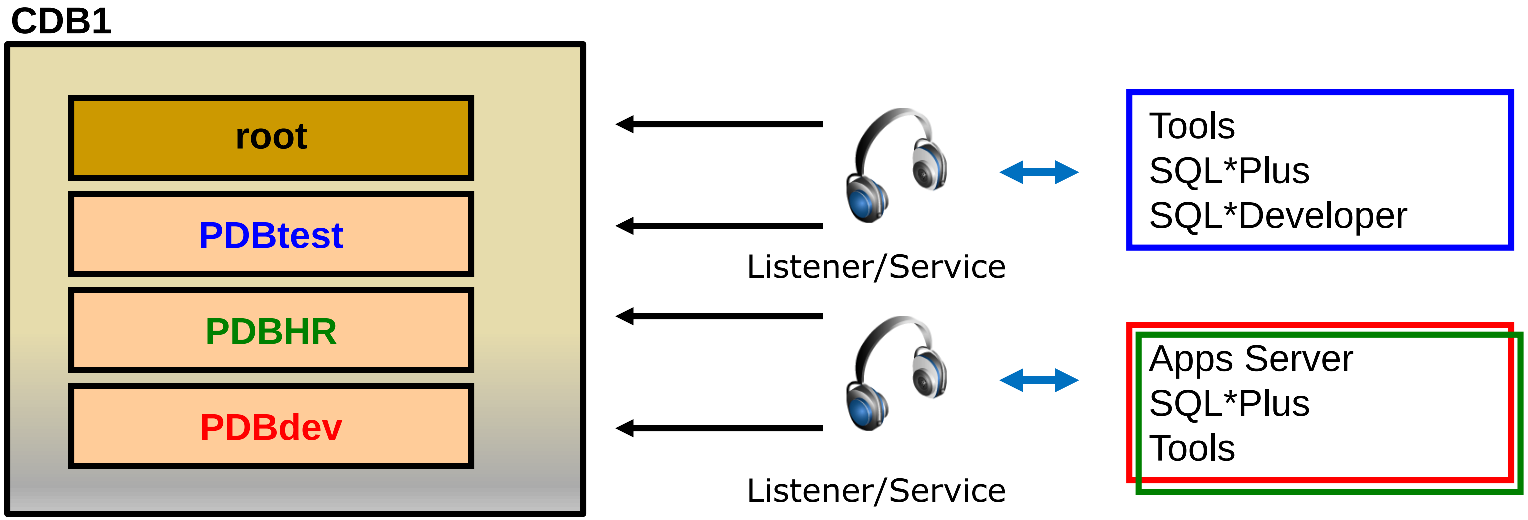
```
SQL> SELECT view_name FROM dba_views WHERE view_name like 'CDB%';
```

- `CDB_pdbs`: All PDBs within CDB
- `CDB_tablespaces`: All tablespaces within CDB
- `CDB_users`: All users within CDB (common and local)

DBA dictionary views providing information within PDB:

```
SQL> SELECT table_name FROM dict WHERE table_name like 'DBA%';
```

Connection to a Non-RAC CDB



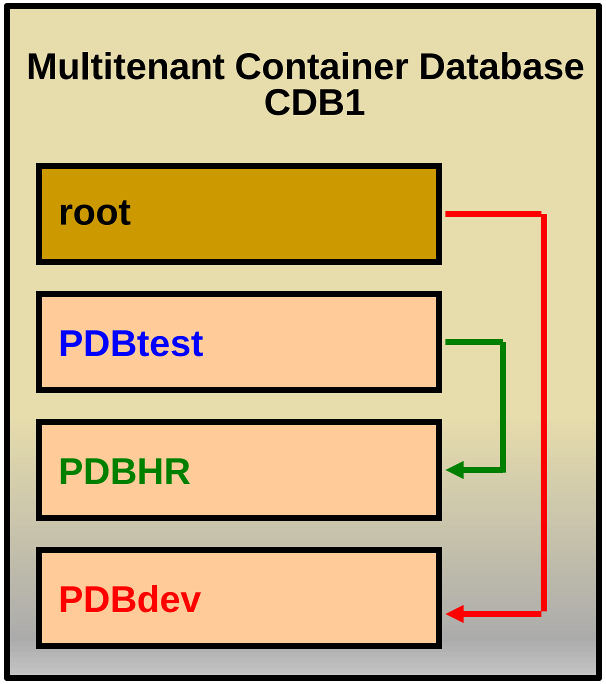
1. Every PDB has a default service created at PDB creation.

```
SQL> SELECT name, pdb FROM cdb_services;
```

2. Service name has to be unique across CDBs.

```
SQL> CONNECT / AS SYSDBA
SQL> CONNECT sys@"hostname:1525/CDB1" AS SYSDBA
SQL> CONNECT sys@"hostname:1525/PDBtest" AS SYSDBA
SQL> CONNECT local_user1@"hostname/PDBHR"
SQL> CONNECT common_user2@"hostname/PDBdev"
SQL> SHOW CON_NAME
```

Switching Connection



Two possible ways to switch connection between containers within a CDB:

- Reconnect

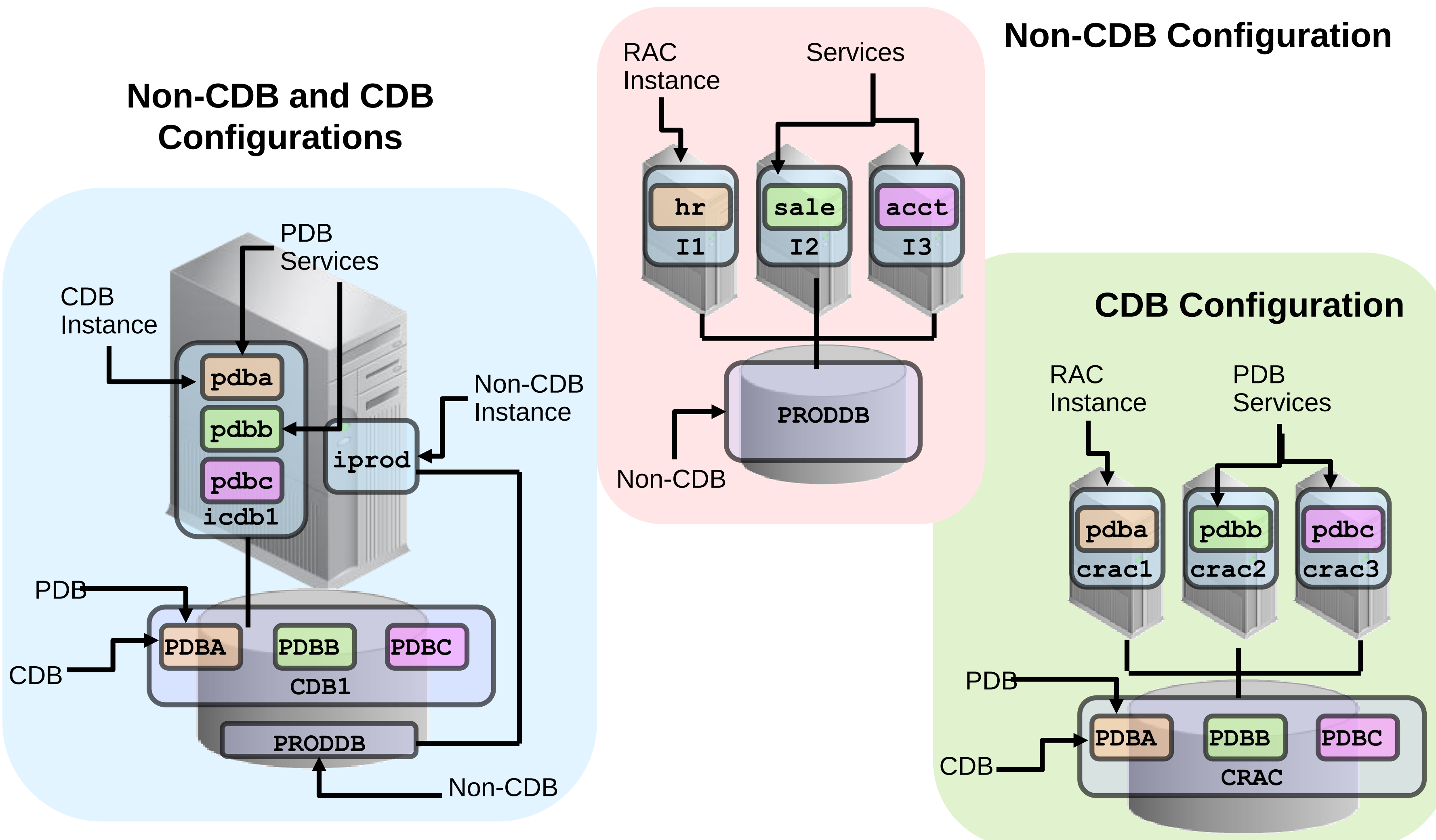
```
SQL> CONNECT / AS SYSDBA
SQL> CONNECT local_user1@PDBdev
```

- Use `ALTER SESSION` statement:

```
SQL> CONNECT sys@PDBtest AS SYSDBA
SQL> ALTER SESSION SET CONTAINER=PDBHR;
SQL> SHOW CON_NAME
SQL> ALTER SESSION SET CONTAINER=CDB$ROOT;
```

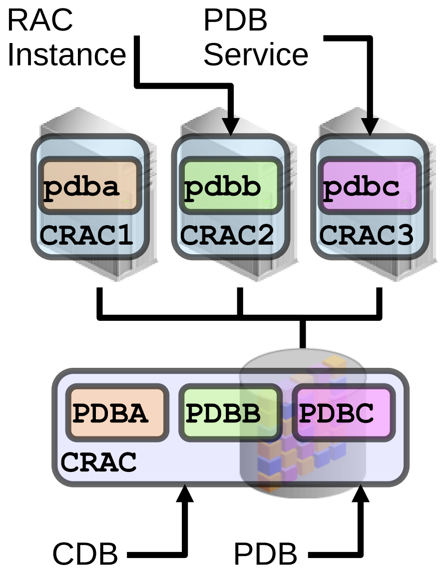
- Using `CONNECT` allows connection under a common or local user.
- Using `ALTER SESSION SET CONTAINER` allows connections only under a common user who is granted the new system privilege, `SET CONTAINER`.
  - `AFTER LOGON` triggers do not fire.
  - Transactions are still pending after switching containers.

Oracle RAC and Multitenant Configuration

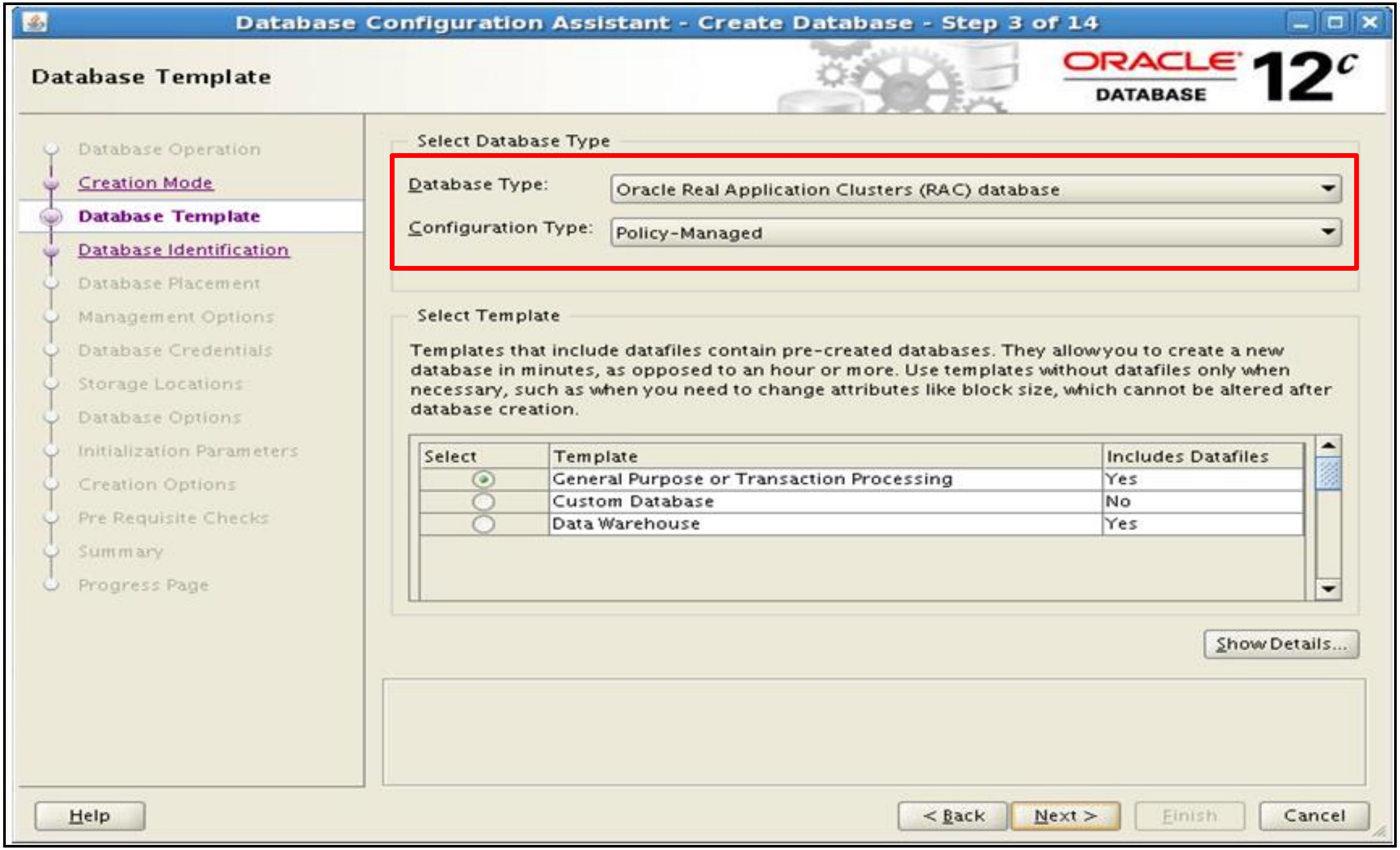


Oracle RAC and Multitenant Architecture

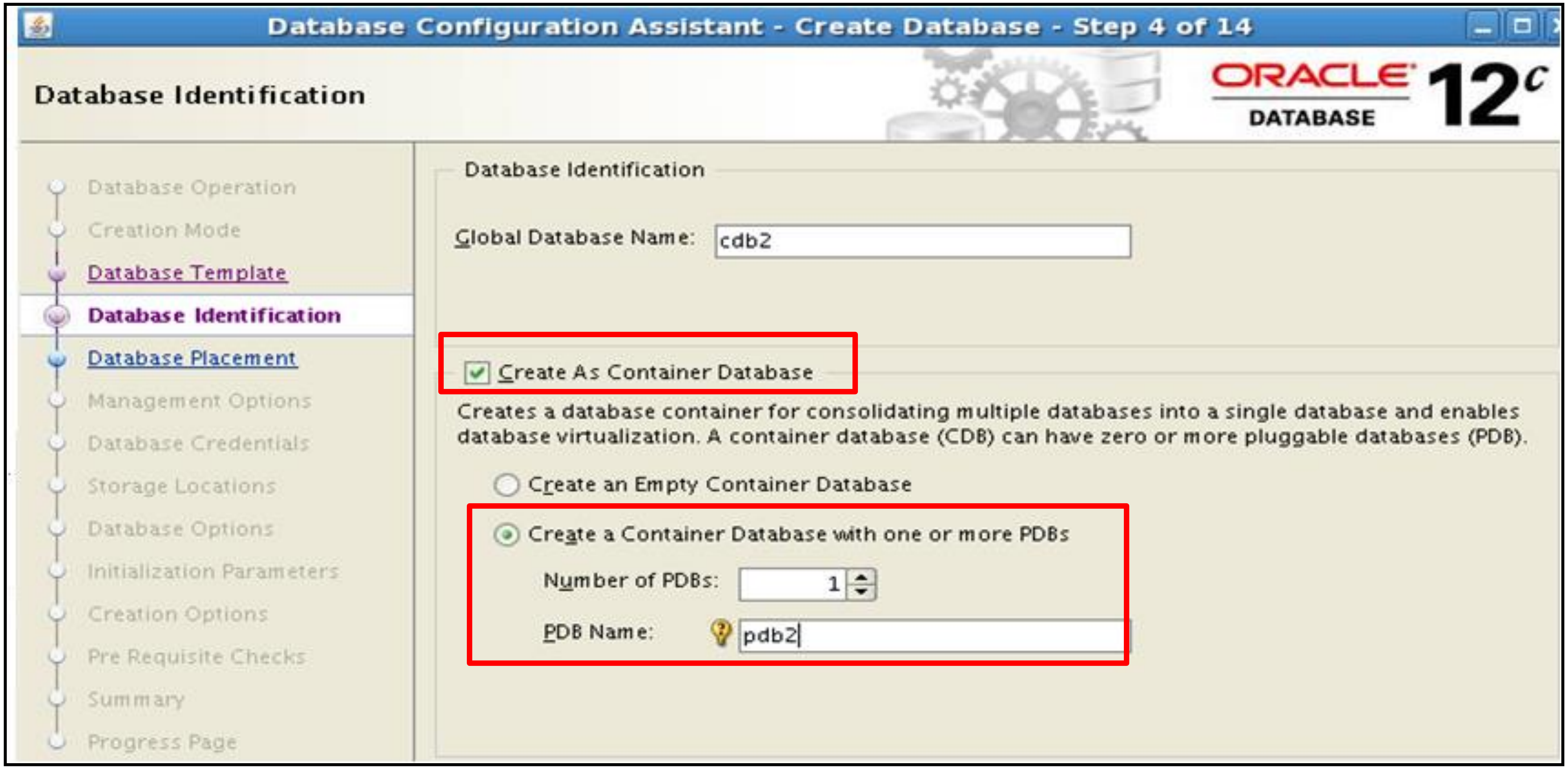
- In a multitenant architecture, there are:
  - Several instances per CDB
  - One UNDO tablespace per CDB instance
  - At least two groups of redo log files per CDB instance
- Each container is exposed as a service:
  - The root
  - Each PDB
- Management of services:
  - Default database services
  - Policy-managed databases:
    - Singletons
    - Uniform across all servers in a server pool



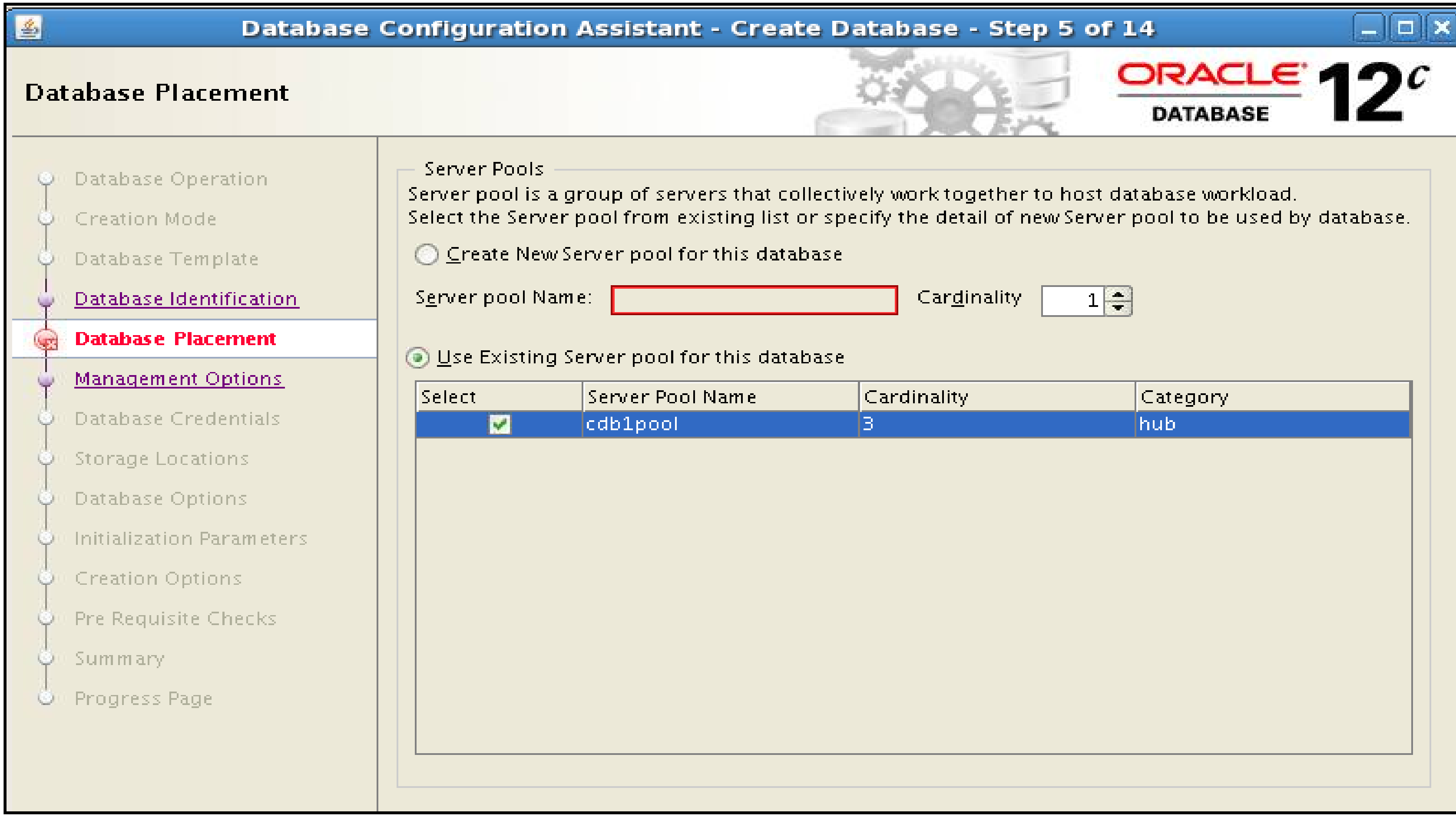
Creating a RAC CDB



Creating a RAC CDB Including PDBs



Hosting a RAC CDB in Server Pools



After CDB Creation

- Check the status of the CDB instances.

```
host01 $ srvctl status database -db cdb2
Instance cdb2_1 is running on node host01
Instance cdb2_2 is running on node host02
Instance cdb2_3 is running on node host03
```

- Check the UNDO tablespaces.

```
host01 $ export ORACLE_SID=cdb2_1
host01 $ sqlplus / as sysdba
SQL> SELECT tablespace_name, con_id FROM cdb_tablespaces
     WHERE  contents='UNDO';

TABLESPACE_NAME                    CON_ID
------------------------------ ----------
UNDOTBS1                                1
UNDOTBS2                                1
UNDOTBS3                                1
```

Connecting Using CDB/PDB Services

- The services for the root and each PDB are started on each node of the server pool: **PDB2** can be accessed on any CDB instance.

```
host01 $ lsnrctl status
…
Service "pdb2" has 1 instance(s).
  Instance "cdb2_3", status READY, has 1 handler(s) for this
service…
```

- To connect to the root container on one of the three CDB instances:

```
SQL> connect system@"host01/cdb2"          ——> root
```

- To connect to a PDB on one of the three CDB instances:

```
SQL> connect system@"host01/pdb2"          ——> PDB
```

Opening a PDB in a RAC CDB

Start the CDB instances.

```
host01 $ srvctl start database -db cdb2
```

1 Open a PDB in the current instance:

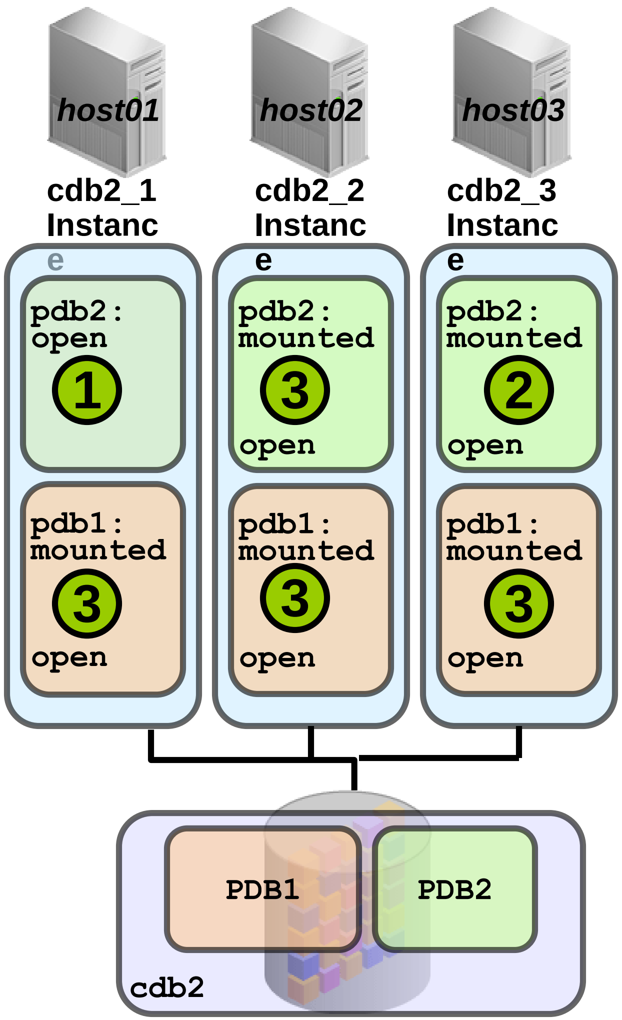
```
SQL> CONNECT sys@host01/cdb2 AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE pdb2 OPEN;
```

2 Open a PDB in some instances:

```
SQL> CONNECT sys@host03/cdb2 AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE pdb2 OPEN
               INSTANCES = ('cdb2_3');
```

3 Open a PDB in all instances:

```
SQL> ALTER PLUGGABLE DATABASE ALL OPEN
                          INSTANCES=ALL;
```



Closing a PDB in a RAC CDB

Shut down the instances of a RAC CDB:

```
host01 $ srvctl stop database -db cdb2
```

- All PDBs closed
- CDB closed/CDB dismounted/Instance shut down

Close a PDB:

- In the current instance only:

```
SQL> CONNECT sys@host01/cdb2 AS SYSDBA
SQL> ALTER PLUGGABLE DATABASE pdb2 CLOSE;
```

- In some or all instances of the CDB:

```
SQL> ALTER PLUGGABLE DATABASE pdb2 CLOSE INSTANCES=('cdb2_3');
SQL> ALTER PLUGGABLE DATABASE pdb2 CLOSE INSTANCES = ALL;
```

- In the current instance and reopen it in another instance:

```
SQL> ALTER PLUGGABLE DATABASE pdb2 CLOSE RELOCATE TO 'cdb2_3';
```

Types of Services

There are two types of services:

- Internal services: `SYS$BACKGROUND, SYS$USERS`
- Application services:
  - A special Oracle CDB service: Created and available by default on all CDB instances
  - PDB (application) services: Limit of 1024 services per CDB

```
SQL> SELECT name, con_id FROM cdb_services;
NAME                 CON_ID
-------------------- ----------
SYS$BACKGROUND                1
SYS$USERS                     1
cdb1XDB                       1
cdb1                          1
pdb1                          3
```

Managing Services

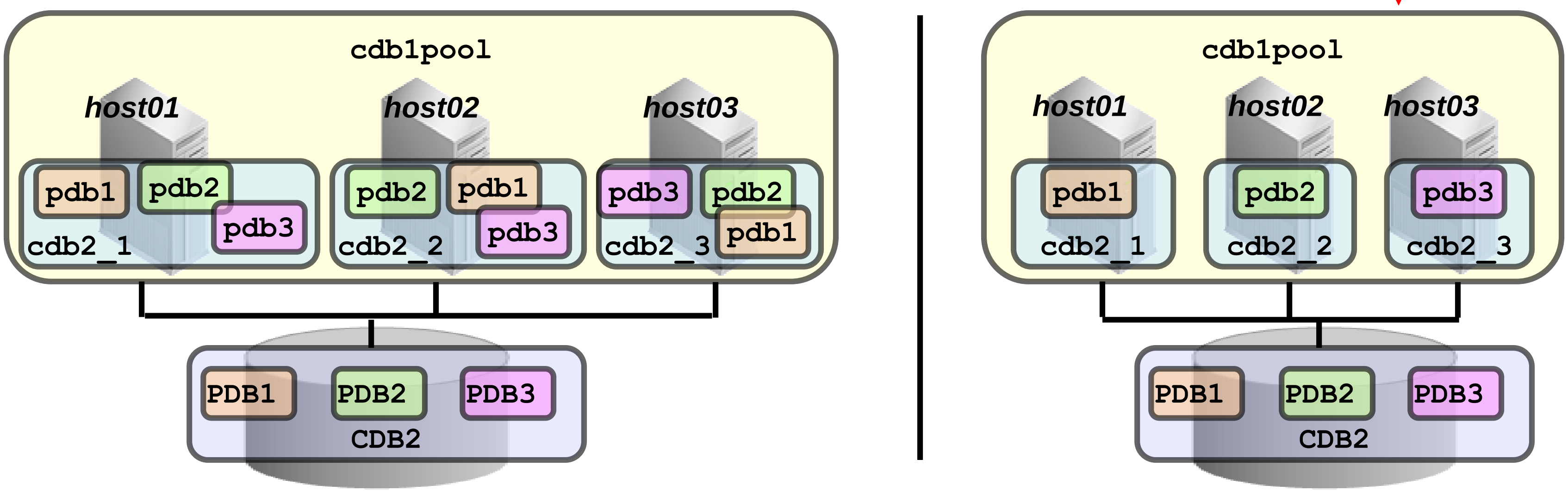
There are two categories of services to manage:

- Default PDB services: one service per PDB
  - Created at PDB creation
  - Not managed by the clusterware
  - Started at PDB opening
- Dynamic PDB services:
  - Useful to start, stop, and place PDBs across the CDB instances
    - Uniformly managed across all nodes in the server pool
    - Running as a singleton service in the same server pool
  - Manually assigned to PDBs with SRVCTL
  - Manually started with SRVCTL
  - Automatically restored to its original state by clusterware
  - PDB automatically opened when the service is started

Affinitizing PDB Services to Server Pools

- A PDB can be assigned several services.
- Each PDB service can be exposed on some or all of the RAC instances within server pools.

```
host01$ srvctl add service -db cdb2 -pdb pdb1 -service pdb1srv
              -policy automatic -serverpool cdb1pool
              -cardinality uniform | singleton
host01$ srvctl start service -db cdb2 -service pdb1srv
```



Adding a PDB to a RAC CDB

Create a new PDB from the seed or clone it from another PDB:

- Connect to the root as a common user with the `CREATE PLUGGABLE DATABASE` system privilege:

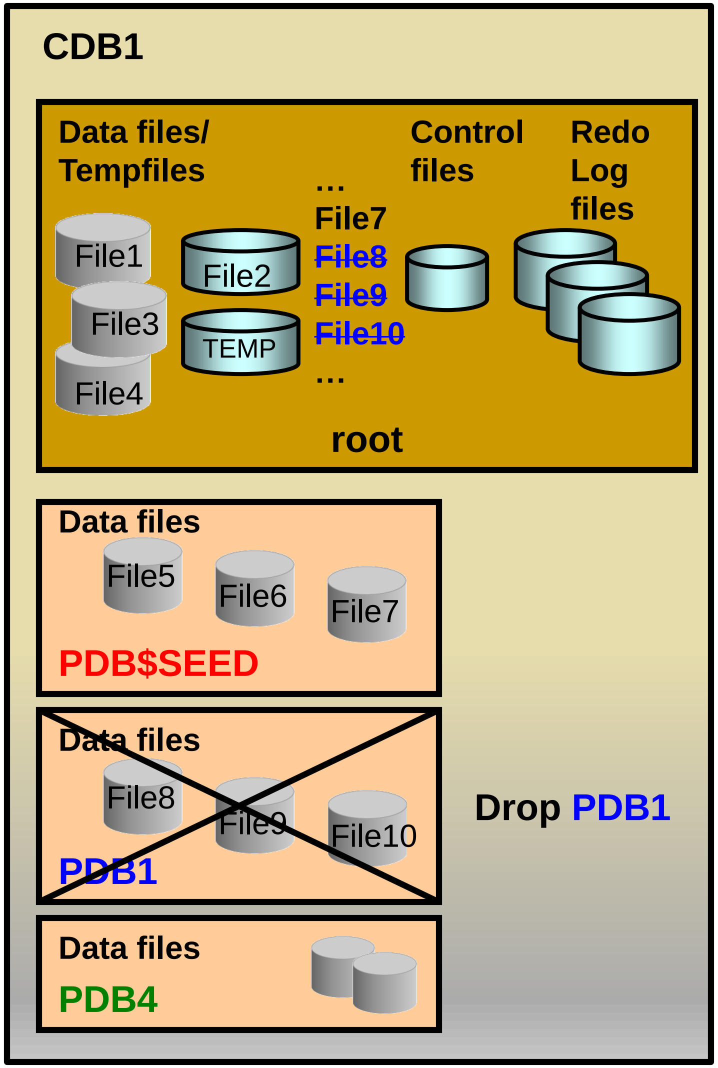
```
SQL> CREATE PLUGGABLE DATABASE pdb1
  2  ADMIN USER admin1 IDENTIFIED BY p1 ROLES=(CONNECT)
  3  FILE_NAME_CONVERT = ('PDB$SEEDdir', 'PDB1dir');
```

```
SQL> CREATE PLUGGABLE DATABASE pdb3 FROM pdb1;
```

- Open the new PDB in the CDB instances.

The default PDB service is automatically started in the CDB instances.

Dropping a PDB from a RAC CDB



1. Remove the dynamic PDB services.

```
host01$ srvctl remove service
               -db cdb1
               -service mypdb1serv
```

2. Close the PDB in all instances.

```
SQL> ALTER PLUGGABLE DATABASE
  2  pdb1 CLOSE INSTANCES=ALL;
```

3. Drop the PDB.

```
SQL> DROP PLUGGABLE DATABASE
  2  pdb1 [INCLUDING DATAFILES];
```

Quiz

Which of the following are true?

a. Only one `SYSTEM` tablespace per CDB
b. Only one instance per PDB
c. A set of redo log files per PDB
d. Only one `UNDO` tablespace per CDB instance
e. One `SYSAUX` tablespace per PDB

Summary

In this lesson, you should have learned how to:

- Describe the multitenant architecture in a non-RAC environment
- Describe the multitenant architecture in a RAC environment
- Create a RAC multitenant container database (CDB)
- Create a pluggable database (PDB) in a RAC CDB
- Use default CDB and PDB services
- Create PDB services
- Associate PDB services with server pools
- Drop a PDB from a RAC CDB

Practice 12: Overview

This practice covers the following topics:

- Creating a RAC CDB with DBCA
- Cloning a PDB in a RAC CDB
- Affinitizing PDB services to CDB instances
- Dropping a PDB from a RAC CDB

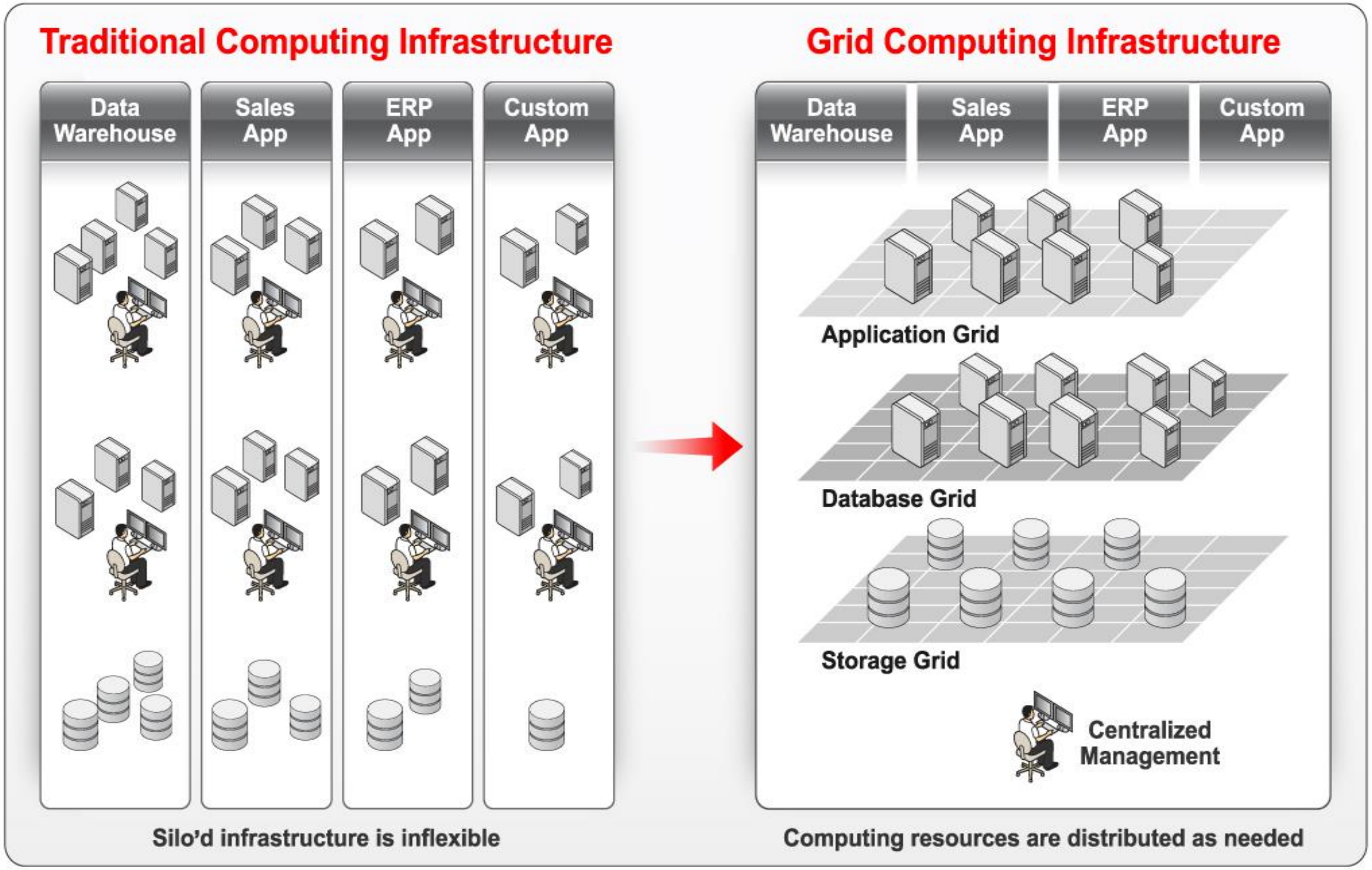
Quality of Service Management

Objectives

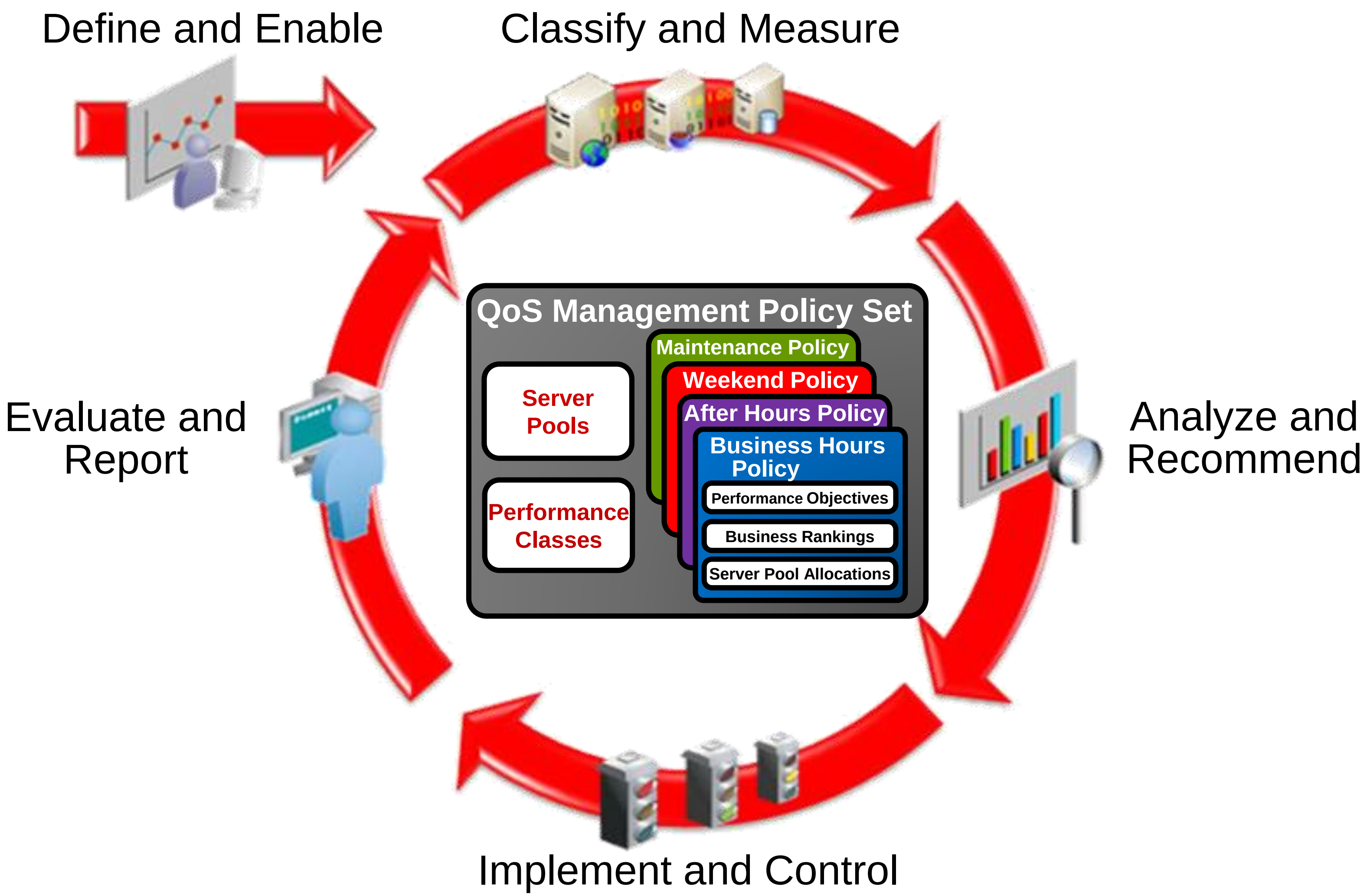
After completing this lesson, you should be able to describe:

- The purpose of Oracle Database Quality of Service (QoS) Management
- The benefits of using Oracle Database QoS Management
- The components of Oracle Database QoS Management
- The operation of Oracle Database QoS Management

QoS Management Background



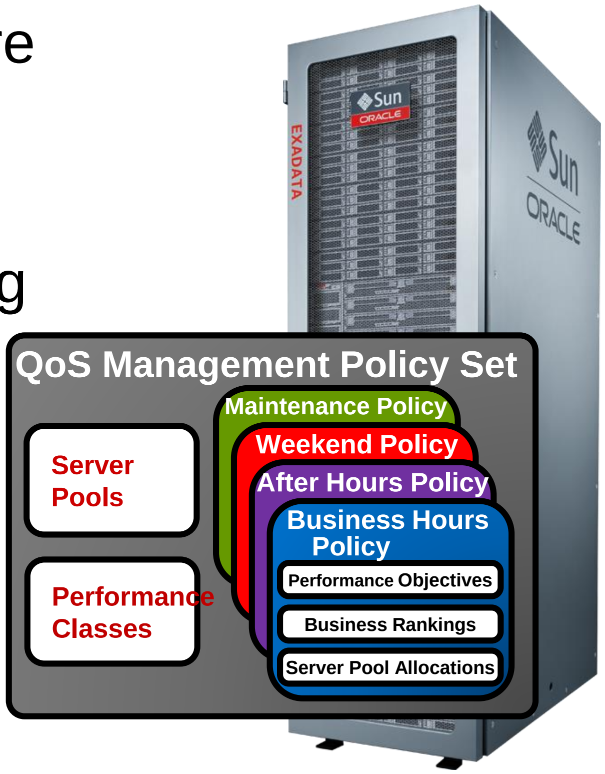
QoS Management Overview



QoS Management and Exadata Database Machine

In its initial form, QoS Management is a feature of the Oracle Database product family:

- Introduced in Oracle Database 11*g* release 2
- Associated with Oracle RAC software
- Released exclusively on Exadata Database Machine
- Focused on environments supporting multiple OLTP workloads
- Not Exadata-specific technology
- The first step along the road towards a broader solution



QoS Management Focus

Resource Use

Code Development
Capacity Planning
Configuration/Deployment

+ Resource Wait

Quality of Service Management

Application Performance

QoS Management Benefits

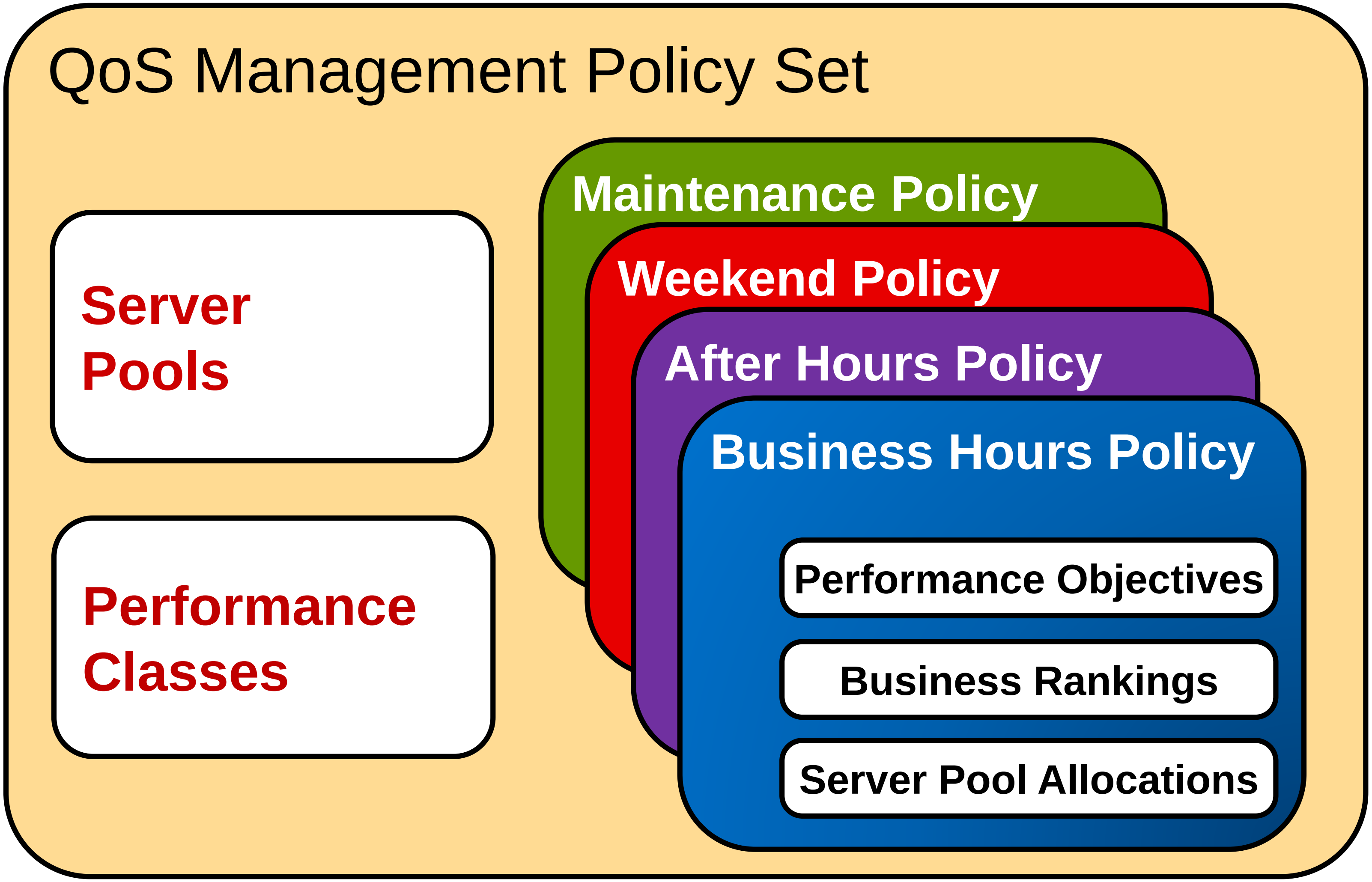
- Determines where additional resources are needed
- Determines whether additional hardware can be added to maintain acceptable performance
- Reduces the number of critical performance outages
- Reduces the time to resolve performance objective violations
- Improves system stability as the workload changes
- Helps to ensure that SLAs are met
- Facilitates effective sharing of hardware resources

QoS Management Functional Overview

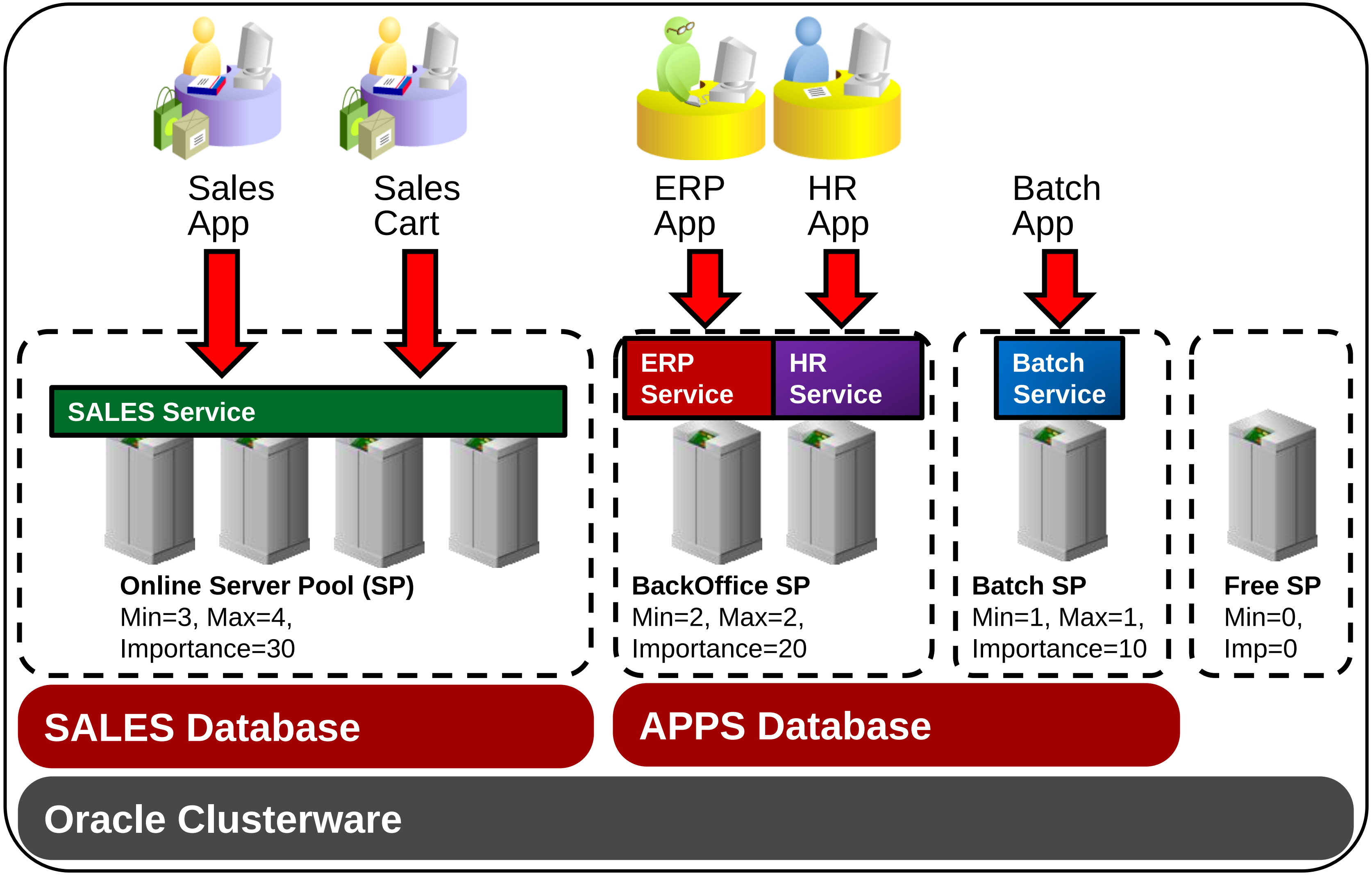
QoS Management works with Oracle RAC and Oracle Clusterware to:

- Manage database server CPU resources by evaluating CPU wait times to identify workloads that are not meeting performance objectives
  - QoS Management can recommend:
    - Adjustments to the size of server pools
    - Alterations to consumer group mappings
    - Adjustments to the CPU resources allocated to different database instances within a server pool
- Manage memory pressure due to number of sessions or runaway workloads
  - QoS Management restricts new sessions from being established on servers that are suffering from memory stress.

QoS Management Policy Sets



Server Pools



Performance Classes

- A performance class is a group of work requests whose service level needs to be managed.
- Work requests are defined by performance classifiers containing the database service name and optional session parameters.
- An initial set of performance classifiers is automatically discovered and created from cluster-managed services.
- Performance objectives are defined on performance classes.

Classification and Tagging

- Each session is classified:
  - The classification is determined by evaluating session parameters against performance class classifiers.
  - Evaluation occurs only when a session is established or when session parameters change.
    - This minimizes the overhead associated with classification.
- Each work request is tagged:
  - The tag is based on the current session classification.
  - The tag connects the work request with a performance class.
    - It enables measurements associated with the work request to be recorded against the appropriate performance class.

Performance Policies

- Performance policies are named sets of performance objectives and server pool overrides to meet business objectives.
  - Performance objectives can be ranked according to their importance.
- Only one policy is active at any time.

Performance Class Ranks

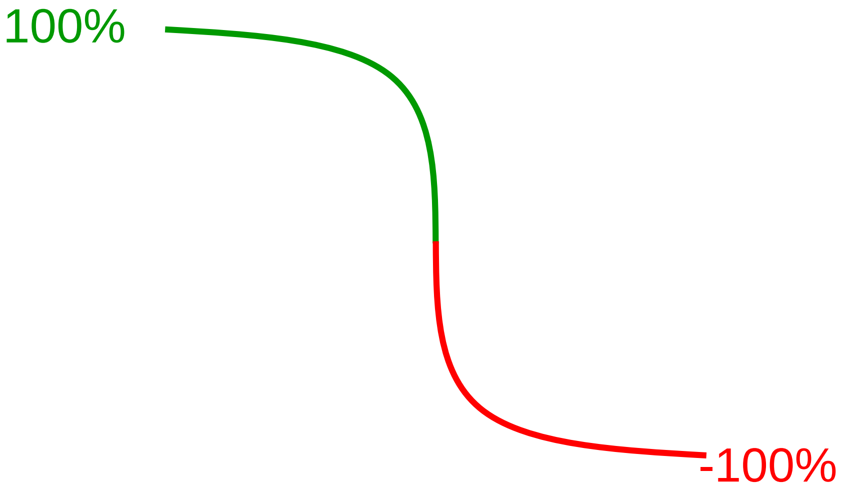
- Performance class ranks assign a relative level of business criticality to each performance class within a performance policy:
  - Highest
  - High
  - Medium
  - Low
  - Lowest

Performance Objectives

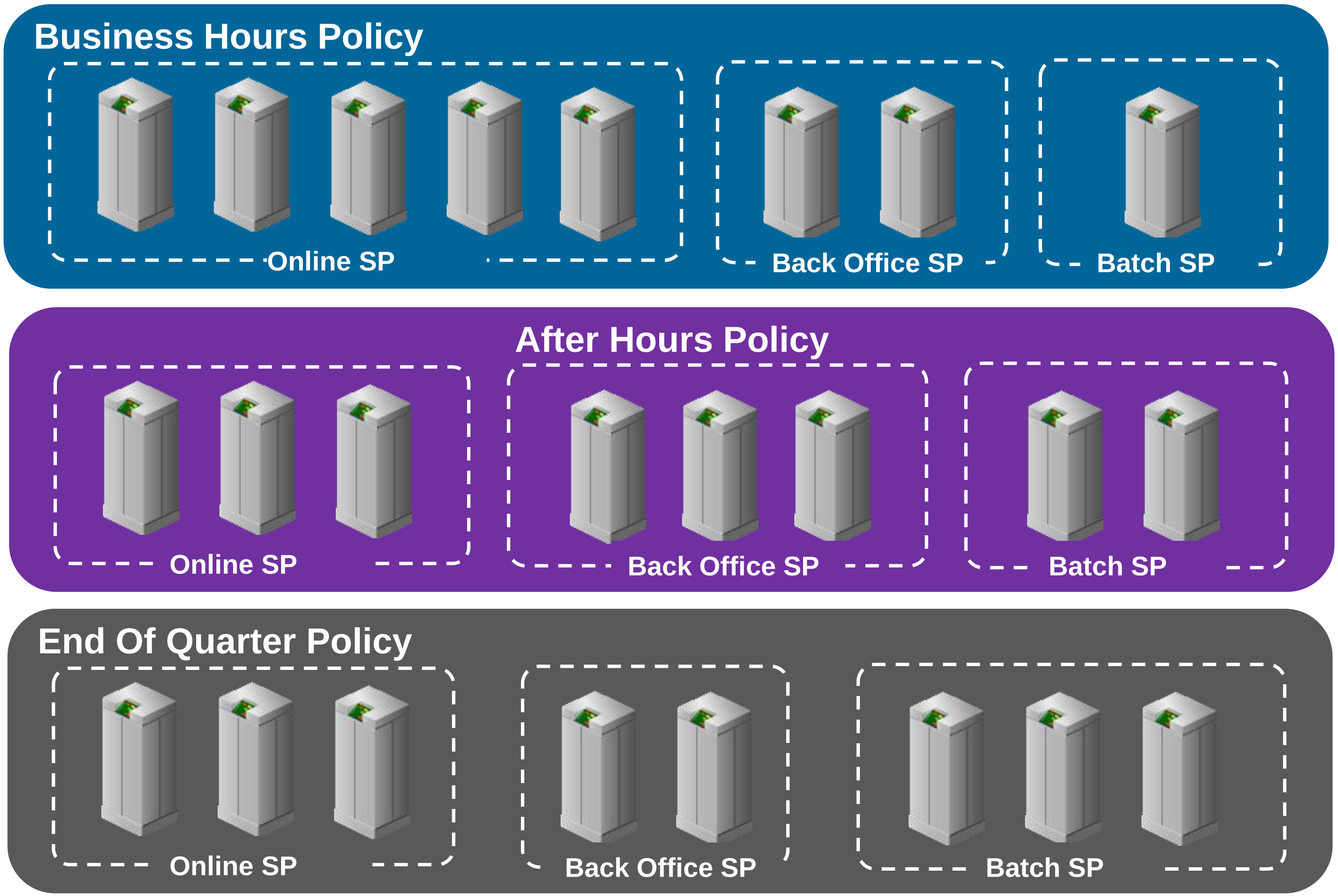
- Performance objectives can be derived from your SLAs. They specify:
  - A business requirement
  - The performance class to which the business requirement applies
- Average response time per database call is currently the only performance objective type.
  - Response time is the total time from the time the database receives the request to when the response leaves the server.
  - Response time does not include network traffic time.

Performance Satisfaction Metrics

- Different performance objectives can be compared using the Performance Satisfaction Metric (PSM).
- The PSM quickly shows how the system is coping with the objective.



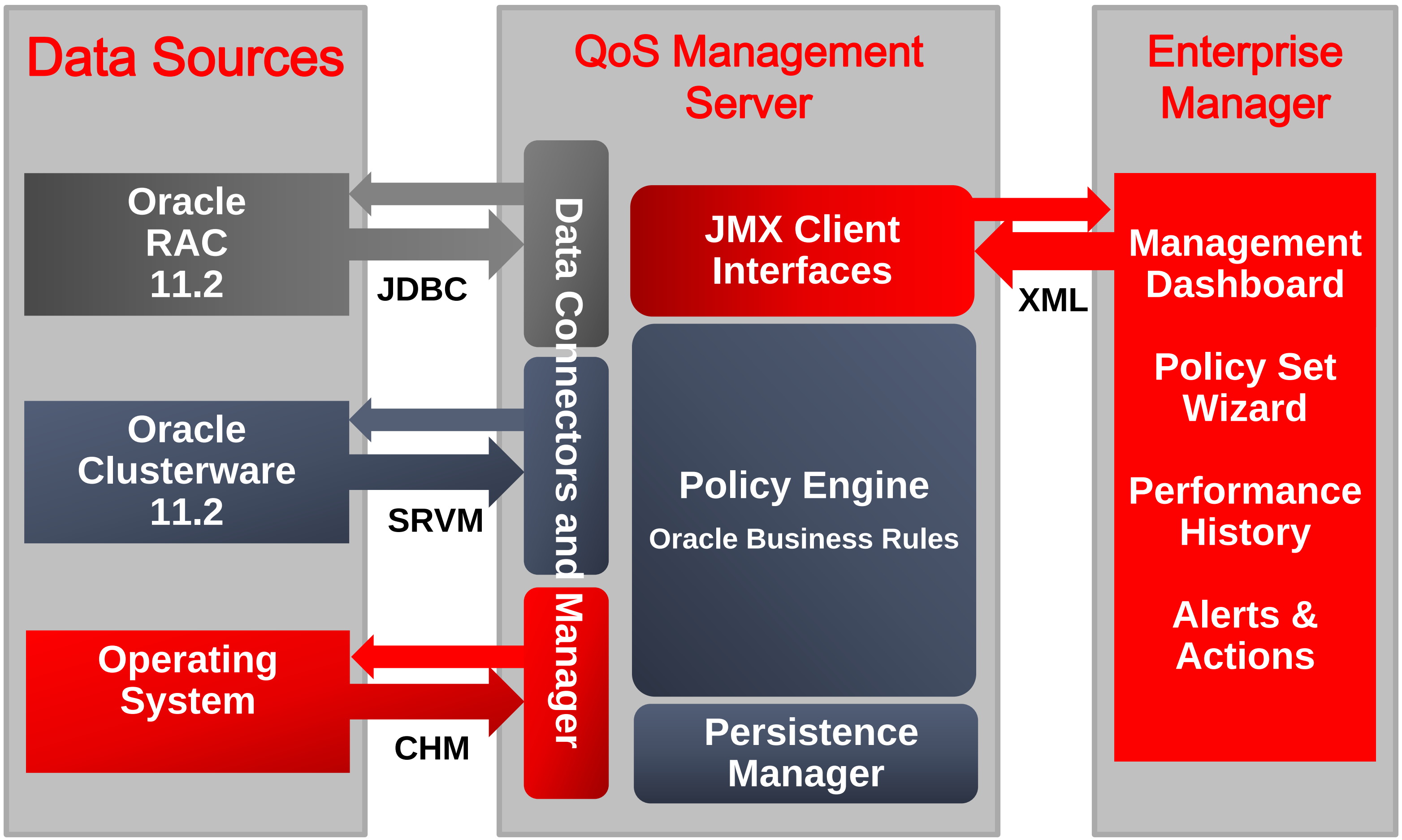
Server Pool Directive Overrides



Overview of Metrics

- QoS Management uses a standardized set of metrics.
- There are two metric types:
  - Performance metrics give an overview of where time is spent in the system.
  - Resource metrics measure the time that work requests use a resource or wait for a resource.
- Metrics are used to identify bottlenecked resources and to determine the best corrective action:
  - For a performance class, the bottlenecked resource is the resource that contributes the largest average wait time.

QoS Management Architecture



QoS Management Recommendations

- If performance objectives are not being met, QoS Management makes a recommendation.
- Each recommendation focuses on improving the highest ranked performance class exceeding its performance objective.
- Recommendations may include:
  - Changing consumer group mappings
    - Reprioritize work within existing resource boundaries.
  - Moving servers between server pools
    - Reprioritize resources between server pools to meet workload demands.
  - Moving CPUs between databases within a server pool
    - Reprioritize CPU resources within existing server pool boundaries.

Implementing Recommendations

ORACLE Enterprise Manager 11*g*
Database Control

Setup Preferences Help Logout

Cluster | Database

Cluster: staiv-cluster1 > QoS Management Dashboard >

Recommended Actions

Action Rank 1: Promote sales cart from Consumer Group 2 to Consumer Group 0.

Action Promote sales cart from Consumer Group 2 to Consumer Group 0.

Estimated Time 2 minutes

Rationale All potential single mapping changes have been analyzed. Changes evaluated and rejected are listed below.

Evaluation The beneficiary's PSM value is expected to change by 3.764 percentage points. The sum of all PSM values is expected to change by -12.314 percentage points. This action is a candidate for recommendation.

Projected Results

	Performance Satisfaction Metric (Last 5 min)		Average Response Time		
Performance Class	Projected (%)	Projected Change (%)	Objective Value (sec)	Current Value (sec)	Projected Value (sec)
Default	100	0.0	0.00000	0.01155	0.01706
hr app	71	0.0	0.01000	0.00293	0.00293
sales app	-84	-16.1	0.00800	0.02504	0.05039
erp app	42	0.0	0.00500	0.00291	0.00291
sales cart	-84	3.8	0.00500	0.04141	0.03157

Implement

Situation Analysis

Donor Performance Classes	Donor Server Pools
Quality of Service Management could help sales cart at the expense of sales app: sales app is another PC using resource cpu in Server Pool online. sales app's Performance Objective is of lesser rank than sales cart's Performance Objective.	Quality of Service Management could move servers from Server Pool backoffice to Server Pool online: The current size of Server Pool backoffice is larger than its configured minimum size.
Quality of Service Management could help sales cart at the expense of Default: Default is another PC using resource cpu in Server Pool online. Default is not currently violating its Performance Objective. Default's Performance Objective is of lesser rank than sales cart's Performance Objective.	

Cluster | Database | Setup | Preferences | Help | Logout

About Oracle Enterprise Manager

Quiz

Oracle Database Quality of Service Management helps to meet performance objectives by reducing resource usage.

a. True
b. False

Quiz

Oracle Database Quality of Service Management recommendations can include:

a. Moving servers between server pools
b. Adding spindles to improve I/O performance
c. Changing consumer group mappings
d. Recommending partitioning strategies to ease global cache bottlenecks

Summary

In this lesson, you should have learned how to describe:

- The purpose of Oracle Database Quality of Service (QoS) Management
- The benefits of using Oracle Database QoS Management
- The components of Oracle Database QoS Management
- The operation of Oracle Database QoS Management

Lesson 13 Demonstrations

- Configuring Quality of Service Management
- Using Quality of Service Management

Oracle Database Exadata Cloud Service Overview

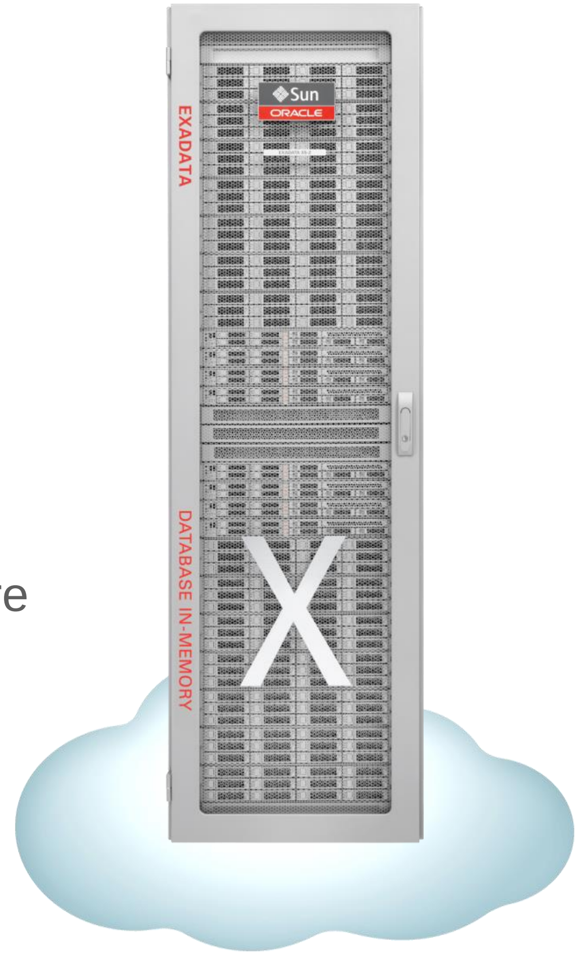
Objectives

After completing this lesson, you should be able to:

- Describe the architecture and capabilities of Exadata Cloud Service
- Compare and contrast between Exadata Cloud Service and an on-premise Exadata implementation

Introducing Exadata Cloud Service

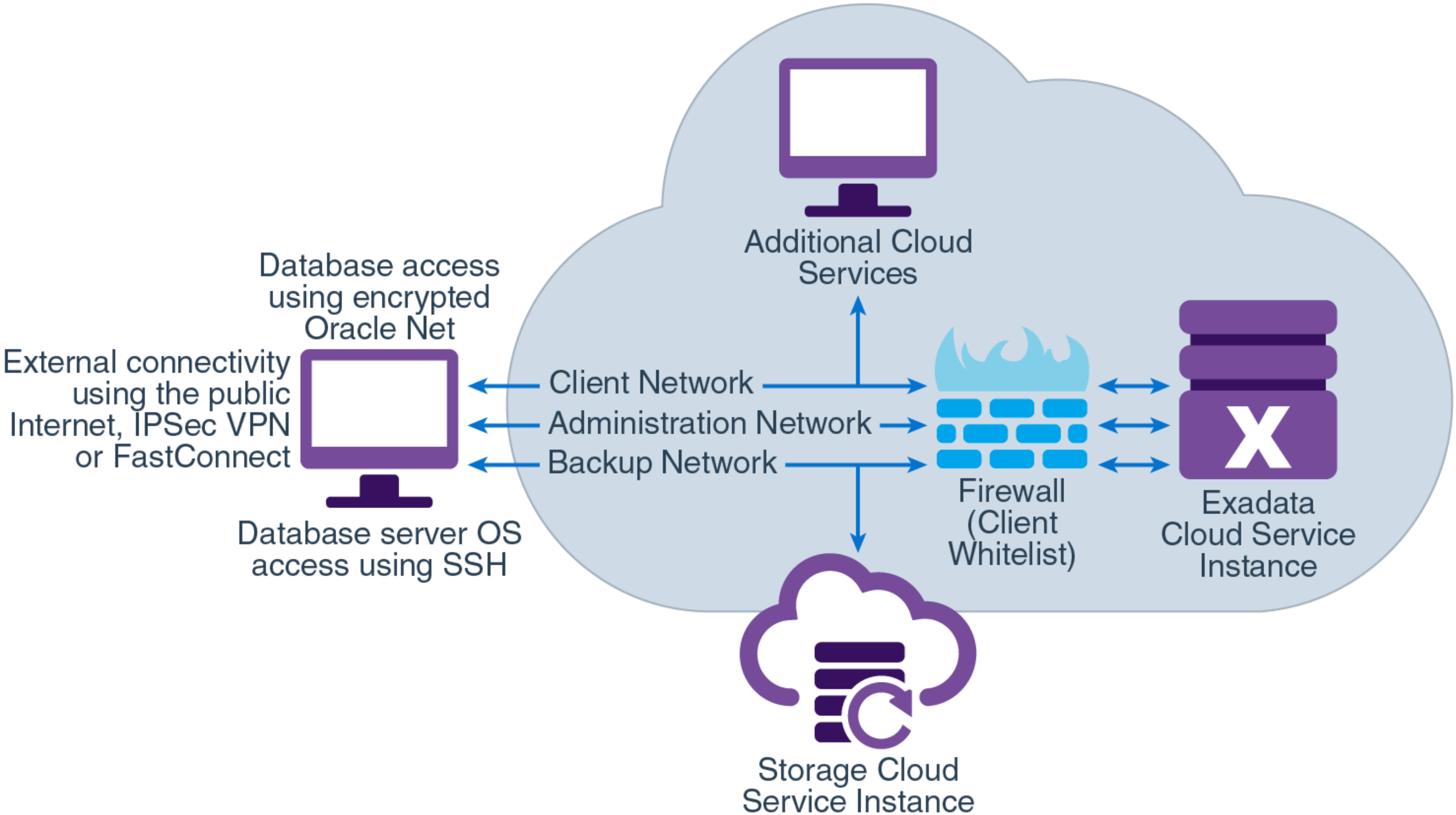
- Oracle Database with all features and options:
  - Industry-leading database for mission-critical OLTP and analytics
- On Exadata Database Machine:
  - The fastest and most available database cloud platform
- In the Oracle Cloud:
  - No capital expenditure, just a simple monthly subscription
  - Oracle deploys and manages the infrastructure
  - Fast, elastic, web-driven service provisioning
  - Complete service isolation with no over-provisioning
  - 100% compatibility with on-premises applications and Oracle database



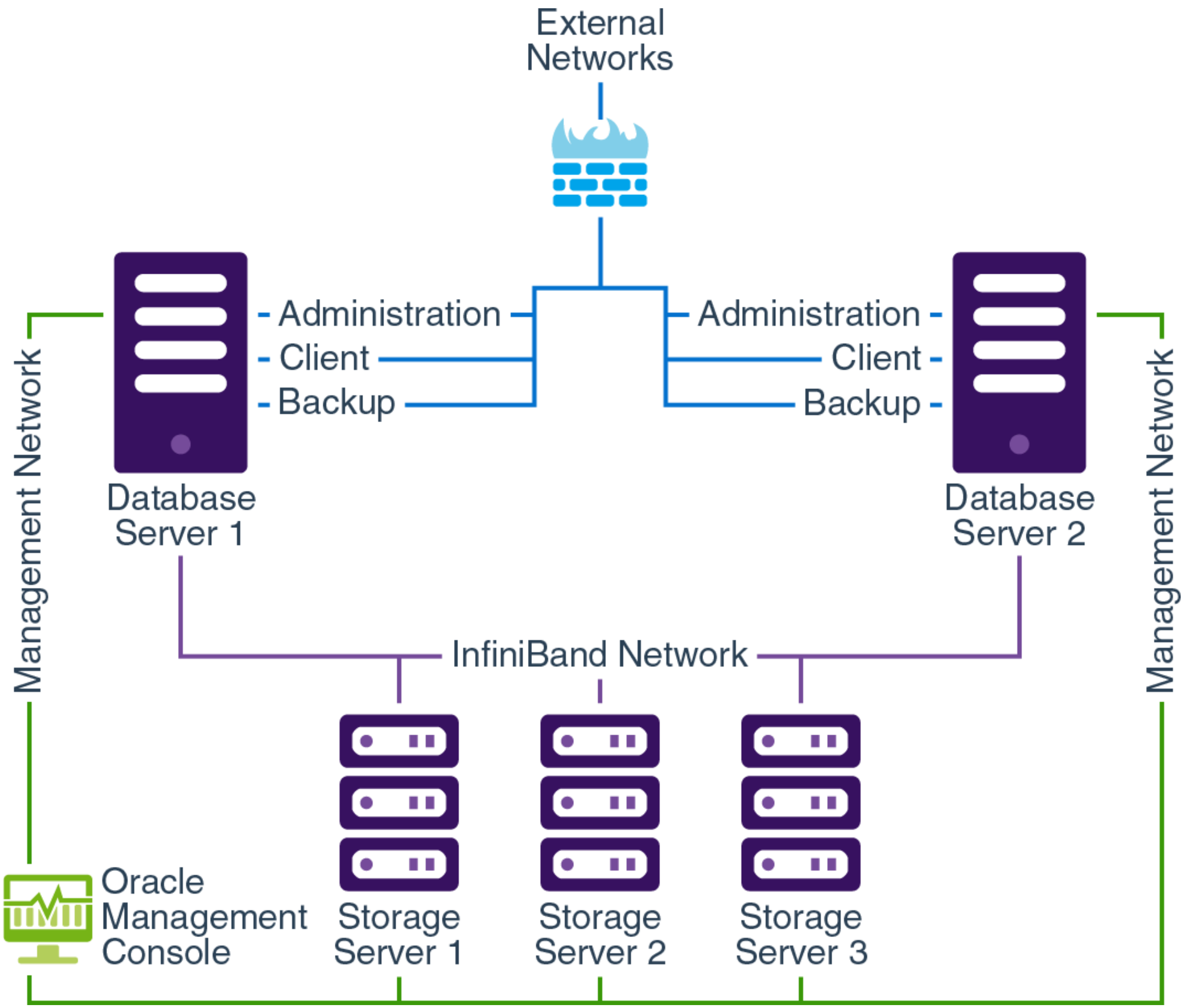
Service Configuration Options

	Quarter Rack	Half Rack	Full Rack
Number of Database Servers	2	4	8
Number of CPU Cores [1]	16 - 68	56 - 136	112 – 272
Total RAM Capacity	496 GB	992 GB	1984 GB
Number of Exadata Storage Servers [2]	3	6	12
Total Flash Capacity	19.2 TB	38.4 TB	76.8 TB
Total Usable Disk Capacity [3]	42 TB	84 TB	168 TB
Maximum SQL Flash Bandwidth [4]	30 GB/sec	60GB/sec	120GB/sec
Maximum SQL Flash Read IOPS [5]	900,000	1,800,000	3,600,000
Maximum SQL Flash Write IOPS [6]	500,000	1,000,000	2,000,000
Maximum SQL Disk Bandwidth [4]	4.5 GB/sec	9 GB/sec	20 GB/sec
Maximum SQL Disk IOPS [5]	7,000	14,000	28,000
Maximum Data Load Rate [7]	5 TB/hour	10 TB/hour	10 TB/hour

Service Connection Options



Service Architecture



Service Availability

- Exadata Cloud Service inherent HA capabilities:
  - Exadata platform full hardware redundancy:
    - Database and storage servers
    - InfiniBand and Ethernet networking
    - Power supplies and PDUs
  - Oracle software HA capabilities:
    - RAC protects against database server failures
    - ASM data mirroring protects against storage failures
      - Exadata Cloud Service used 3-way mirroring (ASM high redundancy)
    - Plus Flashback technologies, Online DDL, In-memory fault-tolerance, and so on...
- Implementation best-practices are derived from thousands of mission-critical deployments worldwide

Management Responsibilities

Oracle Managed (No Customer Access)	Customer Managed (No Oracle Access)
Initial configuration and installation	Database server DomU - including OS
Exadata Storage Server software and objects	Oracle Database, Grid Infrastructure, ASM
Database server Dom0	Database and OS updates*
InfiniBand switches, HCAs, and partitioning	Database and OS monitoring*
Management and ILOM networks	Database backup and recovery*
All hardware, firmware, and BIOS	
Client access VLANS and IP addresses	* Oracle tooling provided

Storage Configuration

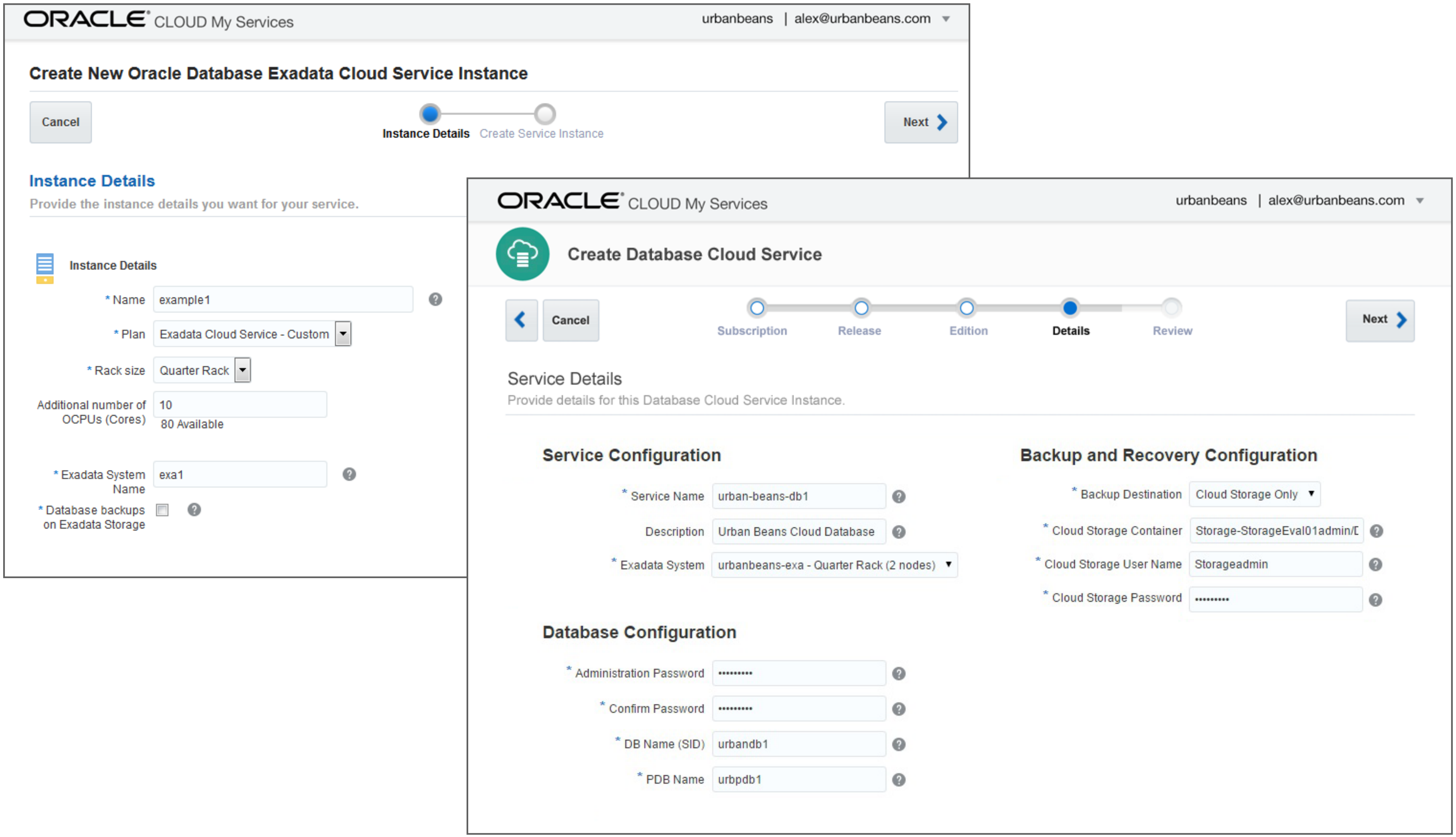
- Preconfigured ASM Disk Groups:
  - High redundancy disk groups, which consume nearly all of the storage
    - DATA contains Oracle Database data files
    - RECO contains the Fast Recovery Area (FRA)
  - System disk groups, which are comparatively very small
    - DBFS contains shared clusterware files
    - ACFS disk groups contain Oracle Database binaries and patch files
- Configuration options for space allocation:
  - Provision for Local Backups: 40% DATA, 60% RECO
  - No Provision for Local Backups: 80% DATA, 20% RECO

	Quarter Rack	**Half Rack**	**Full Rack**
Total Usable Disk Capacity	42 TB	84 TB	168 TB
Disk Group Sizes ***with*** Provision for Local Backups	DATA: 16.8 TB RECO: 25.2 TB	DATA: 33.6 TB RECO: 50.4 TB	DATA: 67.2 TB RECO: 100.8 TB
Disk Group Sizes ***without*** Provision for Local Backups	DATA: 33.6 TB RECO: 8.4 TB	DATA: 67.2 TB RECO: 16.8 TB	DATA: 134.4 TB RECO: 33.6 TB

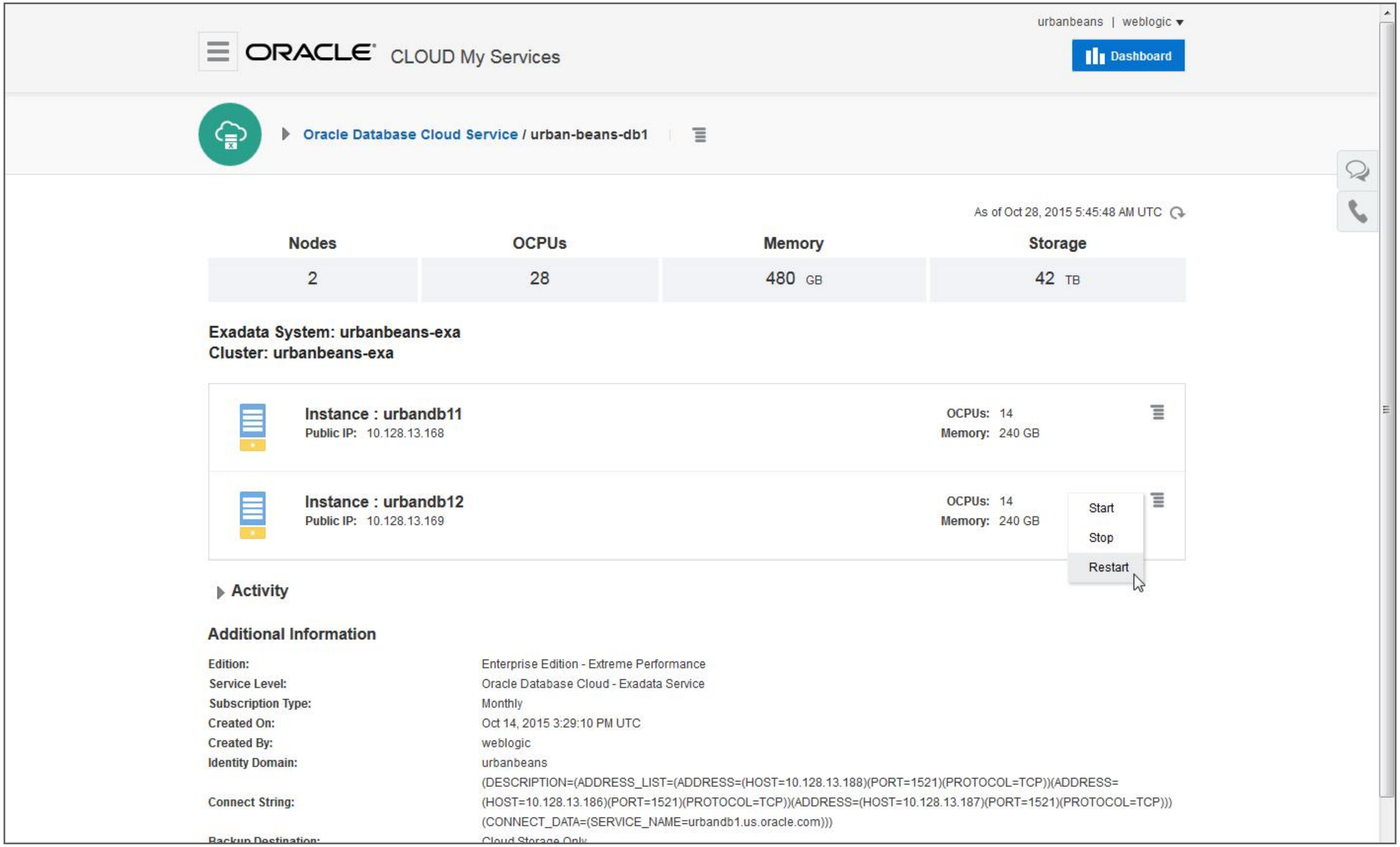
Storage Management Details

- Storage is preconfigured and allocated to ASM disk groups
- Customers manage database objects inside ASM
- Oracle manages the Exadata storage, including Exadata Storage Server software updates
- Exadata Storage Servers are configured using best practices:
  - One cell disk on each physical storage device
  - One set of grid disks for each ASM disk group
    - Space allocation depends on backup configuration
    - Disk groups for database files use high redundancy
    - Grid disk names are prefixed with the corresponding disk group name
  - Flash cache and flash log are preconfigured
    - Flash cache mode is write-through
  - IORM plan is active, IORM objective is set to balanced
- Customers have no direct access to the Exadata Storage Servers
  - Can request custom configurations for grid disks and IORM

Simple Web-Based Provisioning



Simple Web-Based Management

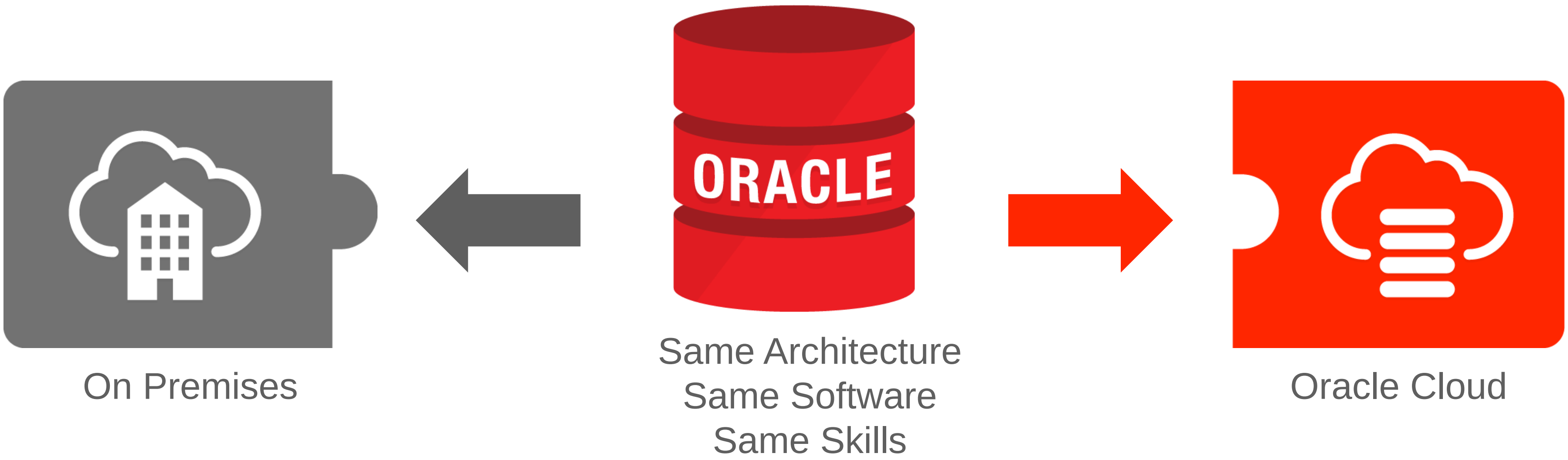


REST APIs

- REST APIs provide programmatic management and control:
  - Create or Delete a Database Deployment
  - Stop, Start or Restart a Database Server VM
  - View Details
    - View a Database Deployment
    - View All Database Deployments
    - View Database Servers
    - View the Status of an Operation

```
$ curl --include --request POST --cacert ~/cacert.pem
--user serviceadmin:Pa55_word
--header "X-ID-TENANT-NAME:usexample”
--header "Content-Type:application/json"
--data '{ "lifecycleState" : "stop", "vmName" : "node02" }`
https://dbaas.oraclecloud.com/paas/service/dbcs/api/v1.1/instances
/usexample/db12c
```

Migrating to Exadata Cloud Service



- Logical Migration options include:
  - Oracle Data Pump
  - Oracle GoldenGate
- Physical Migration options include:
  - Recovery from an RMAN backup
  - Transportable Tablespaces
  - Oracle Data Guard

Summary

In this lesson, you should have learned how to:

- Describe the architecture and capabilities of Exadata Cloud Service
- Compare and contrast between Exadata Cloud Service and an on-premise Exadata implementation